Accurate Three-Axis Control of Spacecraft with Non-Uniform Mass Distribution

Patrick Muñoz Annesen and Manuela Mendes Gonçalves

Master's Thesis, Control and Automation

June 2, 2023

In collaboration with:

SPACE INVENTOR

Control and Automation
Department of Electronic Systems
Aalborg University
es.aau.dk

AALBORG UNIVERSITY
STUDENT REPORT

Title:
Accurate Three-Axis Control of Spacecraft with Non-Uniform Mass Distribution

Theme:
Master's Thesis

Project period:
Feb 2023 - Jun 2023 (Spring semester)

Participants:
Patrick Muñoz Annesen
Manuela Mendes Gonçalves

Supervisor(s):
Henrik Schiøler (AAU)
Ivelin Penchev (Space Inventor Contact)

Pages: 54
Appendix included: 57

Date of completion:
June 2, 2023

This thesis is done in collaboration with Danish satellite manufacturer Space Inventor. A method of performing a transformation allowing for principle axis control in the satellite body was proposed by the company as a way of counteracting the non-homogeneous mass distribution of a satellite. This was attempted using both a PD and LQR approach. Furthermore, a comparison with between the proposed method and applying control directly in the body frame was done through a Monte-Carlo simulation, and their performance was measured with a quadratic cost function. From the results, the principal axis control was deemed to perform no better than the body frame implementation, and the results shown in both case scenarios even favored the body frame over the principal axis controller. A different unsolicited approach presented to Space Inventor was that of Model Reference Adaptive Control (MRAC), which was later attempted. The implementation showed satisfactory reference model tracking and matched uncertainty cancellation in the linear domain. Future work however, revolves a few modifications in order to make the controller also applicable in the nonlinear case.

Contents

Preface

Acknowledgements

A big thanks to Aalborg University's Associate Professor Henrik Schiøler for agreeing to be the supervisor, providing guidance and advice with his wealth of knowledge and years of experience, and thanks to Space Inventor for the collaboration opportunity.

Chapter 1

Introduction

Initially, the purpose of smallsats, mostly CubeSats, was purely to perform in-orbit demonstrations or for educational purposes [10]. However, with technology growing more advanced, companies all over the world uses smallsats as a cheaper alternative with the hope of expanding the possibilities for space exploration. Therefore, the type of mission is a satellite's primary design determinant. If the satellite has to perform numerous in-orbit tasks, it may need additional room for extra components and subsystems, thus becoming larger. Although, since the production and launch costs increases with their size (in weight), large satellites are thus undesirable and, as a result, smallsats become increasingly more popular. SmallSat satellites are classified by their mass, with nanosatellites, cubesatellites and microsatellites being the most prominent [11].

This thesis is made in collaboration with Space Inventor, a Danish satellite company based in Aalborg, manufacturing nano- and microsatellites. Space Inventor's recent endeavors in the satellite industry are considered ground-breaking as they recently, during the writing of this thesis, has been the first satellite company to send a CubeSat made for communication, into a Geostationary orbit [6]. Launched with SpaceX's Falcon Heavy rocket, the 16U CubeSat "G-Space 1" was manufactured on behalf of the company Gravity Space and has the primary task of providing IoT (Internet of Things) communication services. Additionally "G-Space 1" is equipped with an imaging system to capture images of Earth and possibly other celestial objects [3].

Figure 1.1: Image of the 16U CubeSat "G-Space 1", posted by Space Inventor.

Manufacturing satellites having to meet the demands of customers and deliver state-of-the-art solutions with off-the-shelf components, is no easy task. Especially in terms of satellite pointing accuracy. In the words of Space Inventor, given the project proposal in Appendix A, today's standards on spacecraft pointing accuracy are exceedingly higher with the advancements in technology, bringing modern antennas, optical payloads etc. requiring ultra-fine pointing accuracy.

Numerous factors contribute to the challenge of achieving ultra-fine pointing, however, the notion of inertia off-diagonality and non-homogeneous mass distribution, causing unwanted cross-couplings of the satellite axes, is of special interest to Space Inventor.

Cross-coupling is something that can never be avoided in a satellite's inertia tensor, as it is nearly impossible to achieve a fully symmetrical mass distribution, especially if the satellite is carrying rather large fuel tanks. For example, a satellite's inertia tensor could be designed with either full or empty tanks. Regardless, as soon as that fact changes in orbit, the inertia either becomes greater or smaller in the respective axes determined by the placement of the fuel tanks. Additionally, as inertia changes so does the torque needed to rotate the satellite, and thus using a space where the inertia tensor is diagonal, might be more beneficial, as suggested by Space Inventor.

Chapter 2

Preliminary Study

The preliminary study will be comprised of the theory and analyses needed to be able to proceed with the control design of the spacecraft. First off, some relevant reference frames are presented, next a walk-through of the fundamentals of spacecraft attitude dynamics and lastly a brief case analysis describing the thesis problem stated in the introduction.

2.1 Reference Frames

Being a unit-length coordinate system in 3D Cartesian space, a reference frame is a mission specific frame of reference related to the predefined tasks of the satellite [9][15]. These frames can be satellite specific or Earth and celestial body specific. A few of the most common frames are presented in the following.

Earth-Centered Inertial (ECI)

The Earth-centered inertial frame is as the name suggests, an inertial frame of reference. Inertial frames are non-accelerating and non-rotating frames where Newton's laws apply, making them useful for (spacecraft) motion analysis. ECI has its origin in the Earth's center of mass with fixed axes, i.e., not rotating with Earth. Its **Z**-axis is aligned with Earth's rotational axis, **X**-axis pointing towards the *vernal equinox*, i.e., the point of intersection between the equatorial plane and ecliptic plane, and lastly **Y**-axis completing the right-handed coordinate system. ECI is commonly denoted $\{\mathbf{X}_{\Upsilon},\ \mathbf{Y},\ \mathbf{Z}\}$, $\{\mathbf{X}_I,\ \mathbf{Y}_I,\ \mathbf{Z}_I\}$ or simply $\{\mathbf{I}_x, \mathbf{I}_y, \mathbf{I}_z\}$.

Body Reference Frame (BRF)

The BRF is a coordinate frame located in the satellite center of mass, with its coordinate axes representing the satellite's actual orientation in space [15]. For instance, if the satellite was to perform nadir pointing, the BRF axis equipped with the remote sensing component, would have to align with the **Z**-axis of the ORF. For the BRF its axes are denoted $\{\mathbf{B}_{\mathrm{x}},\ \mathbf{B}_{\mathrm{y}},\ \mathbf{B}_{\mathrm{z}}\}$.

Principal Axis Reference Frame (Control Reference Frame, CRF)

When an object rotates about an axis that is not aligned with its principal axis of inertia, it will experience wobbling or oscillations. In contrast, objects rotating about one of its principal axes will perform pure rotations. The principal axis frame has its origin in the CoM of the satellite opposite to the BRF, which in the case of non-uniformity, has its origin at the geometric center. The principal axis frame is denoted $\{\mathbf{P}_{\mathrm{x}},\ \mathbf{P}_{\mathrm{y}},\ \mathbf{P}_{\mathrm{z}}\}$.

Rounding off the section is an image of the frames used in the forthcoming when describing the spacecraft dynamics:

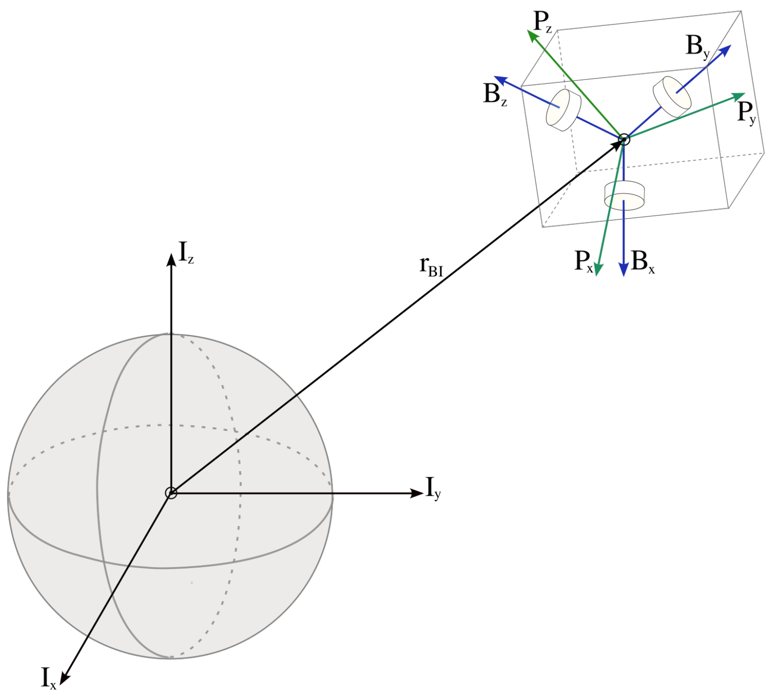


Figure 2.1: Image showing the inertia, body and principal axis references frames.

2.2 Spacecraft Attitude Dynamics Fundamentals

A spacecraft's mission might require pointing of instruments or performing maneuvers. To accomplish those, it is necessary to align its attitude, which is represented by a frame fixed within the body, with a desired frame. The displacement between the spacecraft body frame and the desired frame can be represented by a single rotation using quaternions. [2]

2.2.1 Quaternion Representation

A quaternion can be denoted by a four component vector $\boldsymbol{q} \in \mathbb{H}$, where $\boldsymbol{q_{1:3}}$ is a three-vector part, and q_4 is a scalar [2]

$$\boldsymbol{q} := v_x i + v_y j + v_z k + \eta = [v_x \; v_y \; v_z \; \eta]^\top = [\boldsymbol{v}^\top \; \eta]^\top \tag{2.1}$$

and

$$\begin{aligned} \boldsymbol{v}^\top &= [\sin(\theta/2)\varepsilon_x \;\; \sin(\theta/2)\varepsilon_y \;\; \sin(\theta/2)\varepsilon_z]^\top \\ \eta &= \cos(\theta/2) \end{aligned} \tag{2.2}$$

A unit quaternion defining a rotation from frame A to B can be defined as [2]

$$\boldsymbol{q}_{AB} := [\boldsymbol{v}^\top \; \eta]^\top \in \mathbb{H}_1 := \{\boldsymbol{q} \in \mathbb{H} : |\boldsymbol{v}|^2 + \eta^2 = 1\}. \tag{2.3}$$

The product of two quaternions $\boldsymbol{q}$ and $\bar{\boldsymbol{q}}$ offer two alternatives [2]

$$\begin{aligned} \boldsymbol{q} \otimes \bar{\boldsymbol{q}} &= [\boldsymbol{q}\otimes]\bar{\boldsymbol{q}}, \\ \boldsymbol{q} \odot \bar{\boldsymbol{q}} &= [\boldsymbol{q}\odot]\bar{\boldsymbol{q}}, \end{aligned} \tag{2.4}$$

which relation between the two definitions is

$$\boldsymbol{q} \otimes \bar{\boldsymbol{q}} = \bar{\boldsymbol{q}} \odot \boldsymbol{q}. \tag{2.5}$$

The product $\otimes$ has proved to be more useful in attitude analysis [2]. The next 4×4 matrices are convenient to perform the product between quaternions

$$[\boldsymbol{q}\otimes] := \begin{bmatrix} \eta \mathbf{I}_3 - [\boldsymbol{v}\times] & \boldsymbol{v} \\ -\boldsymbol{v}^\top & \eta \end{bmatrix} = [\Psi(\boldsymbol{q})\boldsymbol{q}] \tag{2.6}$$

$$[\boldsymbol{q}\odot] := \begin{bmatrix} \eta \mathbf{I}_3 + [\boldsymbol{v}\times] & \boldsymbol{v} \\ -\boldsymbol{v}^\top & \eta \end{bmatrix} = [\Xi(\boldsymbol{q})\boldsymbol{q}] \tag{2.7}$$

where $[\boldsymbol{v}\times]$ is the skew-symmetric matrix of a vector $\boldsymbol{v} = [v_x v_y v_z]^\top$ [2]

$$[\boldsymbol{v}\times] := \begin{bmatrix} 0 & -v_z & v_y \\ v_z & 0 & -v_x \\ v_y & v_x & 0 \end{bmatrix}, \quad [\boldsymbol{v}\times]^\top = -[\boldsymbol{v}\times]. \tag{2.8}$$

2.2.2 Error quaternion

A helpful quantity to identify in quaternion-based attitude estimation and control is the error quaternion $\delta\boldsymbol{q}$. For a reference quaternion $\bar{\boldsymbol{q}} \in \mathbb{H}_1$ the relation reads as [2]

$$\begin{aligned} \bar{\boldsymbol{q}} &= \delta\boldsymbol{q} \otimes \boldsymbol{q}, \\ \delta\boldsymbol{q} &= \bar{\boldsymbol{q}} \otimes \boldsymbol{q}^{-1} \end{aligned} \tag{2.9}$$

and by using the cross-product matrix this relation can be rewritten as

$$\delta\boldsymbol{q} = [\bar{\boldsymbol{q}}\otimes]\boldsymbol{q}^{-1} \tag{2.10}$$

For small rotations, the following is a useful first-order approximation, the *small-angle approximation*. The rotations occurring in an infinitesimal portion of time are often expressed in this terms

$$\delta\boldsymbol{q} = \begin{bmatrix} \sin\dfrac{\|\delta\boldsymbol{\theta}\|}{2} \cdot \dfrac{\delta\boldsymbol{\theta}}{\|\delta\boldsymbol{\theta}\|} \\ \cos\dfrac{\|\delta\boldsymbol{\theta}\|}{2} \end{bmatrix} \approx \begin{bmatrix} \dfrac{\delta\boldsymbol{\theta}}{2} \\ 1 \end{bmatrix} \tag{2.11}$$

and given that

$$\boldsymbol{\omega} = \frac{\delta\boldsymbol{\theta}}{\delta t} \tag{2.12}$$

the small-angle approximation can be rewritten as

$$\delta\boldsymbol{q} = \begin{bmatrix} \dfrac{\boldsymbol{\omega}\delta t}{2} \\ 1 \end{bmatrix} \tag{2.13}$$

2.2.3 Attitude Dynamics and Kinematics

The satellite's equations of motion are separated into a kinematic and dynamic model. Kinematics is the study of the satellite time derivative orientation with respect to an inertial reference frame, while the dynamics establish a relationship between the torques affecting the satellite and its angular velocity [2]. One way of deriving the equations of motion for a satellite body is using unit quaternions, since they provide an efficient and compact way for representing orientation and rotations, and provide redundancy which avoid gimbal lock issues.

The quaternion rate of change is given by [2]

$$\dot{\mathbf{q}} = \lim_{\delta t \to 0} \frac{\mathbf{q}(t + \Delta t) - \mathbf{q}(t)}{\delta t} \tag{2.14}$$

with $q(t + \delta t)$ representing the spacecraft's orientation after a time increment δt. This can be expressed as a quaternion product

$$\dot{\mathbf{q}} = \lim_{\delta t \to 0} \frac{\mathbf{q} \otimes \mathbf{q}_r - \mathbf{q}(t)}{\delta t} \tag{2.15}$$

where $\boldsymbol{q}_r$ represents a small rotation defined by the small-angle approximation described in 2.11.

$$\dot{\mathbf{q}} = \lim_{\delta t \to 0} \frac{\mathbf{q} \otimes \begin{bmatrix} \frac{\boldsymbol{\omega} \delta t}{2} \\ 1 \end{bmatrix} - \begin{bmatrix} 0 \\ 0 \\ 0 \\ 1 \end{bmatrix}}{\delta t} = \lim_{\delta t \to 0} \frac{\mathbf{q} \otimes \begin{bmatrix} \frac{\boldsymbol{\omega} \delta t}{2} \\ 0 \end{bmatrix}}{\delta t} \tag{2.16}$$

Thus, the quaternion kinematics with respect to the IRF is defined as [2]

$$\dot{\mathbf{q}}_{IS} = \frac{1}{2} \boldsymbol{\omega} \otimes \boldsymbol{q}_{IS} = \frac{1}{2} \Xi(\boldsymbol{q}) \omega \tag{2.17}$$

The attitude dynamics of a satellite are derived from its *angular momentum*

$$\mathcal{L}_S = \mathbf{J}_S \boldsymbol{\omega}_S \tag{2.18}$$

where

J_S is the inertia tensor of the body

ω_S is the angular velocity of the body

The rate of change of the angular momentum as seen from an inertial reference is then given by the transport theorem

$$\left.\frac{d\mathcal{L}_S}{dt}\right|_I = \left.\frac{d}{dt}\right|_I (\mathcal{L}_x \hat{s}_x + \mathcal{L}_y \hat{s}_y + \mathcal{L}_z \hat{s}_z) \tag{2.19}$$

and by developing this equation we obtain

$$\left.\frac{d\mathcal{L}_b}{dt}\right|_I = \left.\frac{d\mathcal{L}_S}{dt}\right|_S + \vec{\omega}_S \times \mathbf{J}_S \vec{\omega}_S \tag{2.20}$$

It is well known from Newton-Euler equations of motion that

$$\left.\frac{d\mathcal{L}}{dt}\right|_I = \boldsymbol{\tau} \tag{2.21}$$

where $\boldsymbol{\tau}$ represents the total external torques exerted on the spacecraft, e.g. torques from thrusters, solar radiation pressure, air drag, etc. By combining equation 2.20 with 2.21 and isolating the time derivative we obtain the angular acceleration expression as seen from the IRF [2]

$$\dot{\vec{\omega}}_S = -\mathbf{J}_S^{-1}\vec{\omega}_S \times \vec{\omega}_S + \mathbf{J}_S^{-1}\boldsymbol{\tau}_S \tag{2.22}$$

The equations derived above will be used to simulate and evaluate the controller performance in the next chapters.

With the spacecraft kinematics and dynamics in place, a case analysis on its inertia is performed.

2.3 Case Analysis

When an object rotates about an axis not aligned with its principal axis of inertia, it will experience wobbling or oscillations. In contrast, objects rotating about one of its principal axes will perform pure rotations. It is also known that objects with constant angular velocity about their maximum or minimum principal moment of inertia axis will stay dynamically stable, and that rotations about the maximum principal moment of inertia require the minimum kinetic energy the system can have for a specific angular momentum.

The moment of inertia, or rotational inertia of a body, determines the amount of torque needed for a desired angular acceleration about an axis of rotation. This quantity can be described by a rank 2 tensor. The general inertia tensor form might contain off-diagonal elements.

By diagonalizing the general inertia tensor of a body, the resultant matrix yields the principal axis of inertia with the diagonal elements representing the principal moments of inertia. From linear algebra, a square symmetric matrix $\boldsymbol{A}$, can be diagonalized using its eigendecomposition:

$$\boldsymbol{A} = \boldsymbol{Q}\boldsymbol{\Lambda}\boldsymbol{Q}^\top \tag{2.23}$$

With the matrices $\boldsymbol{Q}$ and $\boldsymbol{\Lambda}$ denoting the eigenvectors and eigenvalues of $\boldsymbol{A}$, respectively. In the case of the satellite inertia, the same decomposition can be applied to transform a non-diagonal inertia tensor, to the body's principal axis of inertia:

$$\boldsymbol{J}_b = \boldsymbol{H}\boldsymbol{J}_p\boldsymbol{H}^\top \quad \Rightarrow \quad \boldsymbol{J}_p = \boldsymbol{H}^\top\boldsymbol{J}_b\boldsymbol{H} \tag{2.24}$$

with $\boldsymbol{J}_b$ denoting the non-diagonal inertia tensor of the satellite in the body frame, $\boldsymbol{H}$ its eigenvector matrix and lastly, $\boldsymbol{J}_p$ the diagonal matrix of principal moments. Because of the orthogonality of the eigenvectors $\boldsymbol{H}$ is norm preserving and can thus be considered a rotation in the following way

$$\boldsymbol{H} = \boldsymbol{R}_P^B \quad \Rightarrow \quad \boldsymbol{H}^\top = \boldsymbol{R}_B^P \tag{2.25}$$

with $\boldsymbol{R}_P^B$ and $\boldsymbol{R}_B^P$ signifying a rotation from the principal axis frame to the body frame and vice versa, respectively. The equalities in (2.25) will become a useful tool later on for the initial control design.

2.3.1 Choosing an Appropriate Inertia Tensor

Inspired by [8] who uses an inertia tensor which is assumed to be a low-earth orbit microsatellite, we have considered this an appropriate choice. However, it is chosen to scale it slightly, yielding the following satellite inertia tensor

$$\boldsymbol{J}_b = \begin{bmatrix} 1.4200 & 0.0087 & 0.0136 \\ 0.0087 & 1.7300 & 0.0602 \\ 0.0136 & 0.0602 & 2.0300 \end{bmatrix} \tag{2.26}$$

along with its principal axis inertia tensor calculated according to (2.24)

$$\boldsymbol{J}_p = \begin{bmatrix} 1.4195 & 0 & 0 \\ 0 & 1.7185 & 0 \\ 0 & 0 & 2.0420 \end{bmatrix} \tag{2.27}$$

The inertia tensor in the principal axis is completely free of cross-couplings, hence yielding simpler decoupled dynamics and in turn a simpler control design. For convenience, an open-loop satellite response using both of the inertia tensors is shown below.

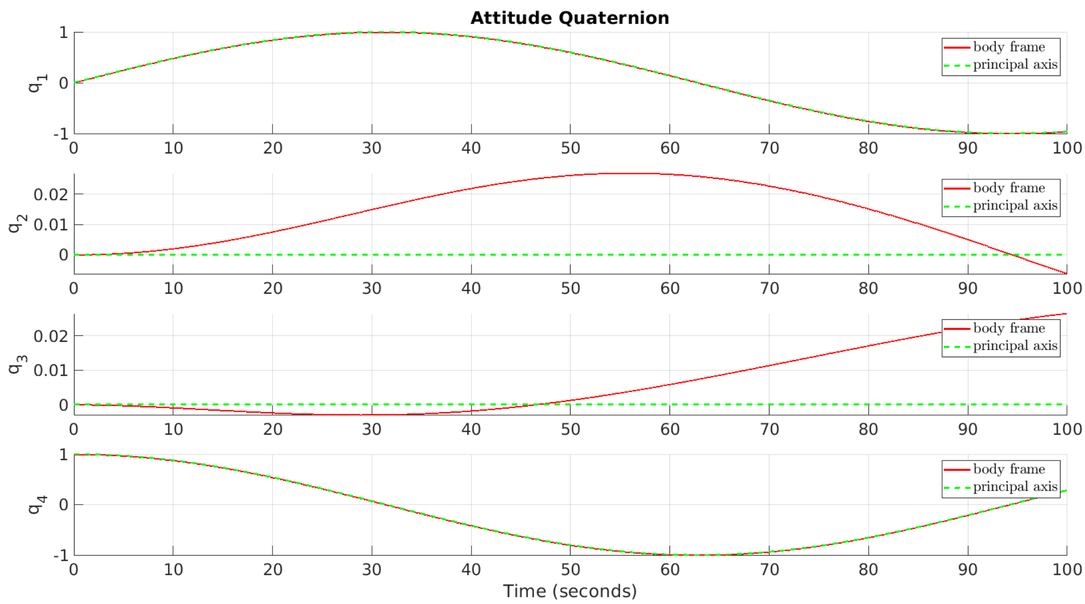


Figure 2.2: Response of the satellite attitude quaternion in open-loop, only initializing the angular velocity to investigate the off-diagonal impact.

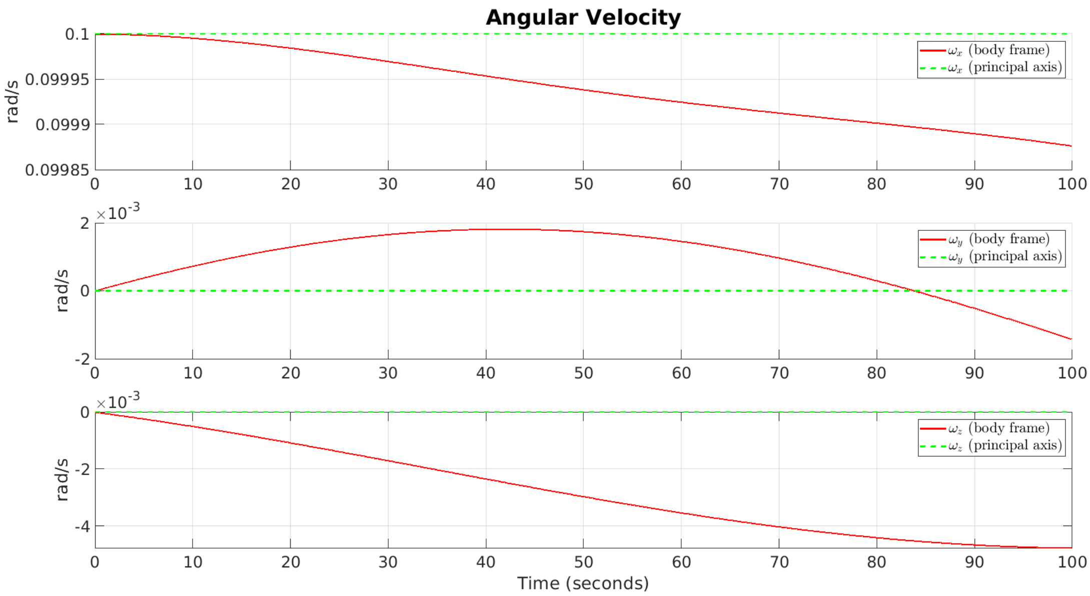


Figure 2.3: Response of the satellite angular velocity in open-loop, initializing the x-axis angular velocity to investigate the off-diagonal impact.

Noting that the satellite has been initialized with spin around its x-axis, namely $\omega_x = 0.1\text{rad/s}$. With the preliminary study concluded, the next chapter will feature an initial attempt on a control design.

Chapter 3

Conventional Attitude Control

In this chapter, the application of a PD controller in two distinct scenarios is examined. The first scenario involves a change of basis from the body frame to the principal axis of inertia, where the control inputs are calculated, and then transformed back to the body frame where the torques are physically applied to the actuators. In the second scenario, the control inputs are calculated directly in the body frame. Finally, the performance of both implementations is compared to assess their effectiveness.

3.1 Controller with Principal Axis of Inertia

In the case of microsatellites with large fuel tanks the mass distribution can be non-homogeneous, which will create cross-coupling dynamics between the satellite body frame axes. In this section, a hypothesis is formulated and tested, where it assumes that by determining the control input in the principal axis first, and then transform it back to the body frame, an improvement of the control performance is achieved. This setup is illustrated in the block diagram of figure 3.1. To that end, it is important to state the relationship between the frame where the control input is determined and the frame where the control input is applied. These frames are the spacecraft body frame and the principal axis frame respectively, denoted as $\mathcal{B}$ and $\mathcal{P}$.

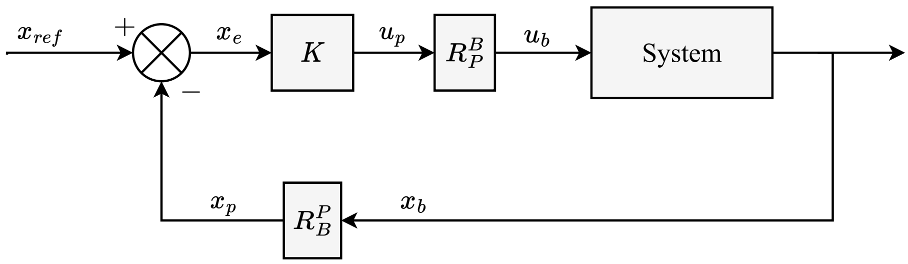


Figure 3.1: Block Diagram of the closed-loop system including the transformations between frames.

A linear transformation defining a rotation from frame $\mathcal{B}$ to $\mathcal{P}$ is given by

$$\boldsymbol{J}_p = \boldsymbol{R}^\top \boldsymbol{J}_b \boldsymbol{R} \tag{3.1}$$

where $\boldsymbol{R}$ is a change of basis matrix from frame $\mathcal{P}$ to $\mathcal{B}$, formed by the eigenvectors of $\boldsymbol{J}_b$, and $\boldsymbol{J}_p$ is a diagonal matrix formed with the eigenvalues of $\boldsymbol{J}_b$ in its diagonal. The desired control inputs are calculated in frame $\mathcal{P}$ and then converted back to frame $\mathcal{B}$ where they will be applied to the reaction wheels.

The control inputs, here represented as external torques τ, satisfy the following relation

$$\tau_B = \boldsymbol{R}\tau_P \tag{3.2}$$

which can be used to transform the torques from one basis to the other.

While working with quaternions it is convenient to express the rotation defined by $\boldsymbol{R}$ in quaternion form. The conversion between a rotation matrix and a unitary quaternion expressing the same rotation is described in Chapter 2. The resultant quaternion is denoted as q_B^P, which can be used to change between body and principal axis frames as follows

$$q_P = \left[q_B \otimes\right] q_B^P \tag{3.3}$$

with

$$q_B := q_I^B \quad \text{and} \quad q_P := q_I^P \tag{3.4}$$

Similarly, the spacecraft angular velocity can also be converted from the body frame to the principal axis frame by

$$\omega_P = \boldsymbol{R}^\top \omega_B \tag{3.5}$$

with ω_1 being the angular velocity expressed in the body frame and ω_2 being the angular velocity expressed in the principal axis.

3.1.1 Control Law

The control law allows determining the poles locations in a closed-loop system to obtain a desired dynamic response. The control law used in the controller design is given by

$$u = \boldsymbol{K} x_e \tag{3.6}$$

Where $\boldsymbol{K}$ is the design parameter, x_e is a vector of state errors, and u is the vector of inputs to the system.

The error state vector x_e gives the difference between a given reference and the current estimated state. For the attitude quaternion this difference is given by the relation [2]

$$q_2 = q_{ref} \otimes q_e \tag{3.7}$$

and by isolating the q_e we obtain

$$q_e = q_{ref} \otimes q_2^{-1} \tag{3.8}$$

for the angular velocity, the ω_e vector is obtained by

$$\omega_e = \omega_{ref} - \omega_2 \tag{3.9}$$

The control inputs are first calculated in the principal axis frame

$$u_P = \boldsymbol{K} x_{eP} \tag{3.10}$$

and then by the relation given in equation 3.2, they are transformed back to the body frame where they will be applied to the reaction wheels.

We propose to design parameter $\boldsymbol{K}$ based on two different approaches, a Linear Quadratic Regulator (LQR) and a Pole Placement approach. Both are linear control strategies, thus requires the model of the system to be linearized.

3.1.2 Linearization

The nonlinear kinematics and dynamics equations governing the behavior of the satellite represented by

$$\dot{x} = f(x, u) \tag{3.11}$$

can be approximated by a Taylor expansion to the first order, which is valid for small deviations from an operating point. For operating points x_0 and u_0, we define the deviations of the system as $\bar{x} = x - x_0$ and $\bar{u} = u - u_0$. The linearized system is then given by [1]

$$f(x, u) \approx \mathcal{T}[f(x, u)] = f(x_0, u_0) + \boldsymbol{A}\bar{x} + \boldsymbol{B}\bar{u} \tag{3.12}$$

where

$$\boldsymbol{A} = \left.\frac{\partial f}{\partial x}\right|_{x_0, u_0} \quad \text{and} \quad \boldsymbol{B} = \left.\frac{\partial f}{\partial u}\right|_{x_0, u_0} \tag{3.13}$$

which yields the following linear model around an operating point [1]

$$\frac{\partial}{\partial t}\begin{bmatrix} \bar{q}_{1:3} \\ \bar{\omega} \end{bmatrix} = \underbrace{\begin{bmatrix} \mathbf{0}_3 & \frac{1}{2}\boldsymbol{I}_3 \\ \mathbf{0}_3 & \mathbf{0}_3 \end{bmatrix}}_{\boldsymbol{A}} \begin{bmatrix} \bar{q}_{1:3} \\ \bar{\omega} \end{bmatrix} + \underbrace{\begin{bmatrix} \mathbf{0}_3 \\ \boldsymbol{J}^{-1} \end{bmatrix}}_{\boldsymbol{B}} \bar{u} \tag{3.14}$$

where $\mathbf{0}_3$ is a 3×3 zeros matrix, $\boldsymbol{I}_3$ is a 3×3 identity matrix, and $\boldsymbol{J}^{\mathbf{-1}}$ is the inverse of the inertia matrix.

This model will be used in the controller design.

3.1.3 PD

The Proportional-Derivative (PD) controller is the most common feedback control used in industry, since a trial-and-error design can be more convenient than advanced control techniques in many applications.Thus, we develop a PD controller to control the 3-axis of the spacecraft for precise pointing. The PD control can be expressed in a state space structure as [14]

$$u(t) = \boldsymbol{K}_p(q_{ref}(t) - q(t)) + \boldsymbol{K}_d(\omega_{ref}(t) - \omega(t)) \tag{3.15}$$

which can be rewritten as

$$u(t) = \boldsymbol{K}(x_{ref}(t) - x(t)) \tag{3.16}$$

where $x(t) = \begin{bmatrix} q(t)_{1:3} & w(t) \end{bmatrix}^T$, and $\boldsymbol{K} = \begin{bmatrix} \boldsymbol{K}_p & \boldsymbol{K}_d \end{bmatrix}$

The proportional gain $\boldsymbol{K}_p$ determines the speed of the system response. A higher K_d will cause the response of the control loop will to reach the reference faster, which might cause overshoot or even make the system go unstable. While the differential term $\boldsymbol{K}_d$ acts on the rate of change of the error state variable, and increasing it will cause the control system to react faster to changes in the error term. The derivative response is sensitive to noise in the state variables, therefore it is common practice to use a small differential term. [1]

The controller gains $\boldsymbol{K}_p$ and $\boldsymbol{K}_d$ are designed by trial-and-error using *Pole Placement*. This design method consists in choosing control gains such that the poles of the closed-loop system are placed on the left half-plane. In state space form, the closed-loop system becomes [1]

$$\dot{x} = (\boldsymbol{A} + \boldsymbol{B}\boldsymbol{K})x \tag{3.17}$$

with the poles being equivalent to the eigenvalues of the matrix $\boldsymbol{A}_{cl} = \boldsymbol{A} + \boldsymbol{B}\boldsymbol{K}$, which should be negative to make the system stable.

The tuning of the PD controller will be explained in *future section*.

3.1.4 Linear Quadratic Regulator

The LQR control problem consists is in minimizing a cost function J, which is expressed as the integral of the square of the states x plus the square of the control inputs u; i.e., [1]

$$J = \int_{t_0}^{t_f} [x^\top(t)\boldsymbol{Q}x(t) + u^\top(t)\boldsymbol{H}u(t)]dt \tag{3.18}$$

with $\boldsymbol{Q} = \boldsymbol{Q}^\top \succeq 0$ and $\boldsymbol{H} = \boldsymbol{H}^\top \succ 0$.

The main question to the control design is the selection of the weights of the matrices $\boldsymbol{Q}$ and $\boldsymbol{H}$. The quadratic form of $x^\top\boldsymbol{Q}x$ impose a penalty or cost associated with deviations of the state x from the origin, and similarly the term $u^\top\boldsymbol{H}u$ represents a cost associated with the control inputs, intended to limit its magnitude so the control signals generated are achievable by the actuators and the control signal saturation occurs at the maximum signal the actuators can produce. Saturation can cause a system to become unstable, thus the control signal weighting matrix should be selected to avoid saturation under normal operation conditions. [1]

The $\boldsymbol{H}$ and $\boldsymbol{Q}$ matrices are designed according to the worse values that the inputs and deviations of the states can achieve, this is known as the Bryson's Rule in literature. The maximum attainable input is given by $\tau_{max} = 0.1Nm$, according to the Space Inventors momentum wheels specification for microsatellites. The matrix penalizing the inputs is then multiplied by some constant α according to the system performance to avoid saturation of the actuators. The maximum value of any components of a unitary quaternion is $q_{max} = |1|$. Thus the $\boldsymbol{H}$ and $\boldsymbol{Q}$ matrices are given by [1]

$$\boldsymbol{H} = \alpha\tau_{max}^{-2}\boldsymbol{I}_3 \quad , \quad \boldsymbol{Q} = \begin{bmatrix} \boldsymbol{I}_3 & 0_3 \\ 0_3 & \boldsymbol{I}_3 \end{bmatrix} \tag{3.19}$$

The K matrix can be then obtained by solving the algebraic Riccati equation with the linearised model.

3.2 Controller with Full Inertia Matrix

In order to test the hypothesis proposed in section 3.1, it is necessary to conduct a comparison to determine whether applying the Body Frame/Principal Axis transformation leads to enhanced performance compared to directly applying the controller to the Body Frame. Thus, a PD and LQR controller were implemented directly with the full inertia matrix.

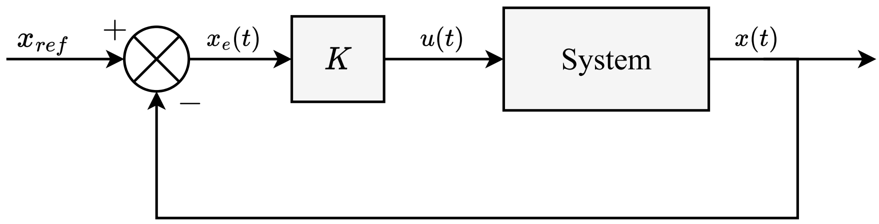


Figure 3.2: Block Diagram of the closed-loop system.

The control inputs are then directly calculated in the Body Frame

$$u_B = \boldsymbol{K}x_{eB} \tag{3.20}$$

where x_{eB} is the state vector given by

$$x_{eB} = \begin{bmatrix} q_{e(1:3)} \\ \omega_e \end{bmatrix} \tag{3.21}$$

and q_e represents the attitude quaternion difference between the current estimated attitude and a reference q_1, and ω_e represents the angular velocity error between the current angular velocity and a given reference ω_1.

$$
\begin{aligned}
q_e &= [q_{ref} \otimes] q_1^{-1} \\
\omega_e &= \omega_{ref} - \omega_1
\end{aligned}
\tag{3.22}
$$

The method used to find the gains $\boldsymbol{K}$ for the PD and LQR controllers follows as the description given in section 3.1. Thus, the main difference between the Principal Axis of Inertia controllers and Body Frame controllers is the inertia matrix used in each case, referring back to Section 2.3 for value of these.

In the next section, the results obtained from both the Principal Axis and Body Frame implementations are presented and examined.

3.3 Comparison Between Principal Axis and Body Frame Controller

The results obtained from both controllers, PD and LQR, applied in the Principal and Body Frame are presented below. Firstly, the controllers are initially compared within their respective categories, where the Body Frame/Principal Axis transformation response with the PD controller is compared, followed by a comparison of the Body Frame/Principal Axis transformation using the LQR controller. Subsequently, their corresponding responses are plotted together to facilitate a comprehensive comparison.

To assess the performance of the controllers under uncertain conditions, a Monte Carlo simulation consisting of 1000 trials was setup. The objective is to test the impact of uncertainties in the inertia matrix and evaluate how effectively the controller can operate. To achieve this, the inertia matrix was varied by ±20% in each simulation, and the controller's performance is then evaluated across a range of possible scenarios.

3.3.1 PD

Evaluating the difference in angular velocity errors between the Body Frame and Principal Axis is challenging due to their closely matched performances, making it difficult to determine a definitive differentiation between the two.

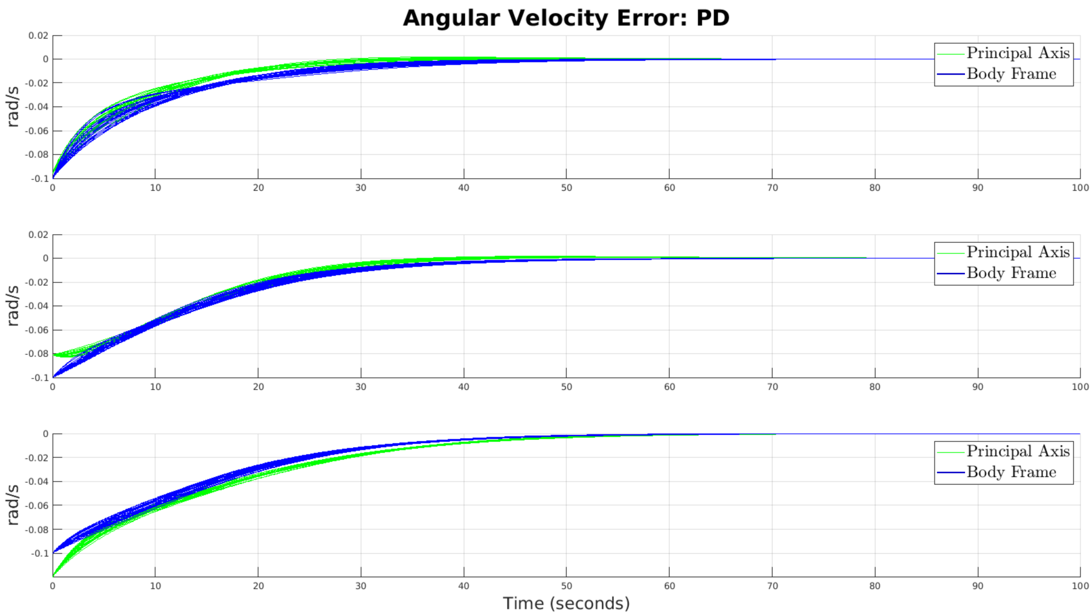


Figure 3.3: Monte Carlo simulation of the angular velocity error for the PD controller applied in the Principal Axis of Inertia and Body Frame.

The scalar part of a quaternion represents a rotation angle, while the vector part determine an axis of rotation. The angle misalignment in the graph below is obtained by taking the arccossine of the scalar part of the quaternion error q_e. And it represents the angle difference between a given reference and the current estimated attitude quaternion. The resultant angle misalignment between both controllers is minor.

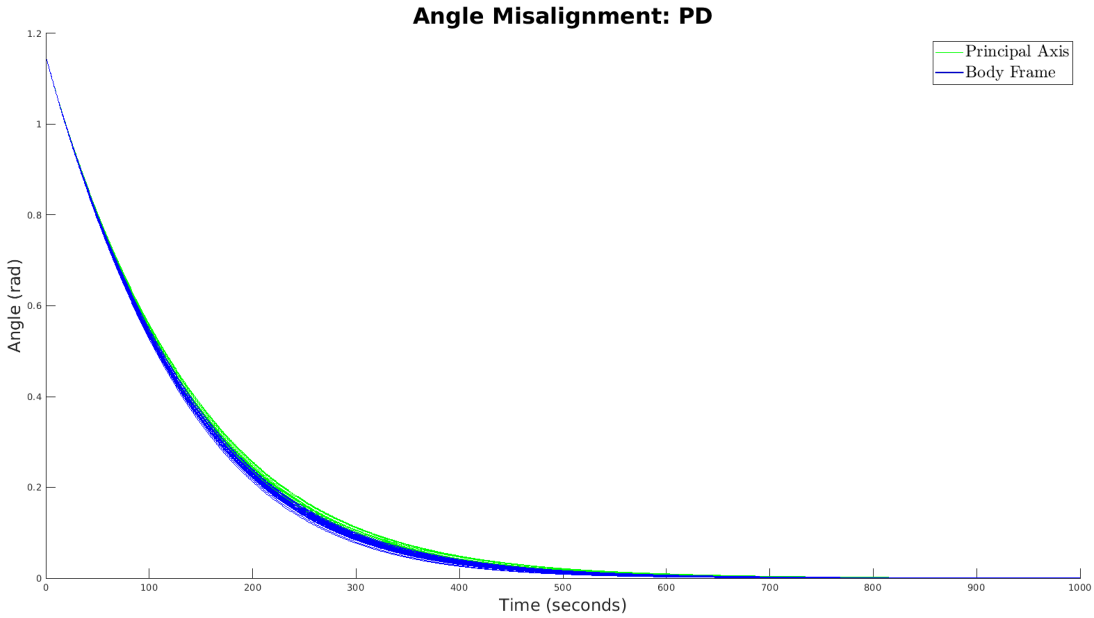


Figure 3.4: Monte Carlo simulation of the angle misalignment between the current attitude estimation and reference.

Throughout all simulated scenarios, both control inputs remain within the saturation limit. While the principal axis requires less torque than the body frame in the second plot, the exact opposite occurs in the third plot.

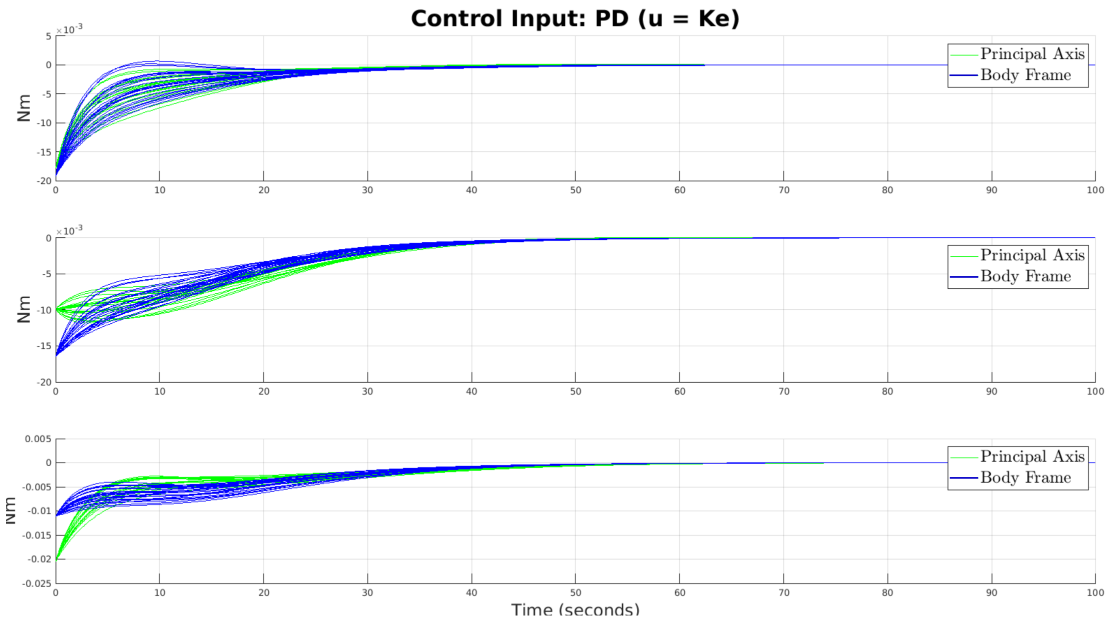


Figure 3.5: Monte Carlo Simulation of the PD control inputs.

Therefore, due to their close performance, it is challenging to determine weather there is an improvement in the system's performance by applying the principal axis transformation, and if so, the extend of its significance. To address this issue, a cost function was then used to obtain a more accurate estimation of both Principal Axis and Body Frame implementations. The cost function is given by

$$P(t) = P(t-1) + (q_{e1:3}^{\top} \cdot \boldsymbol{Q} \cdot q_{e1:3} + u^{\top} \cdot \boldsymbol{R} \cdot u) \cdot dt \tag{3.23}$$

where $\boldsymbol{Q}$ and $\boldsymbol{R}$ are the same matrices used in the LQR implementation. At the end of every Monte Carlo simulation a the cost function yields a scalar that assesses the amount of error accumulated in the states and control inputs, thus indicating the controller performance. A lower value indicates better performance and smaller amount of error.

Based on the obtained values of $\boldsymbol{P}$ a histogram was created

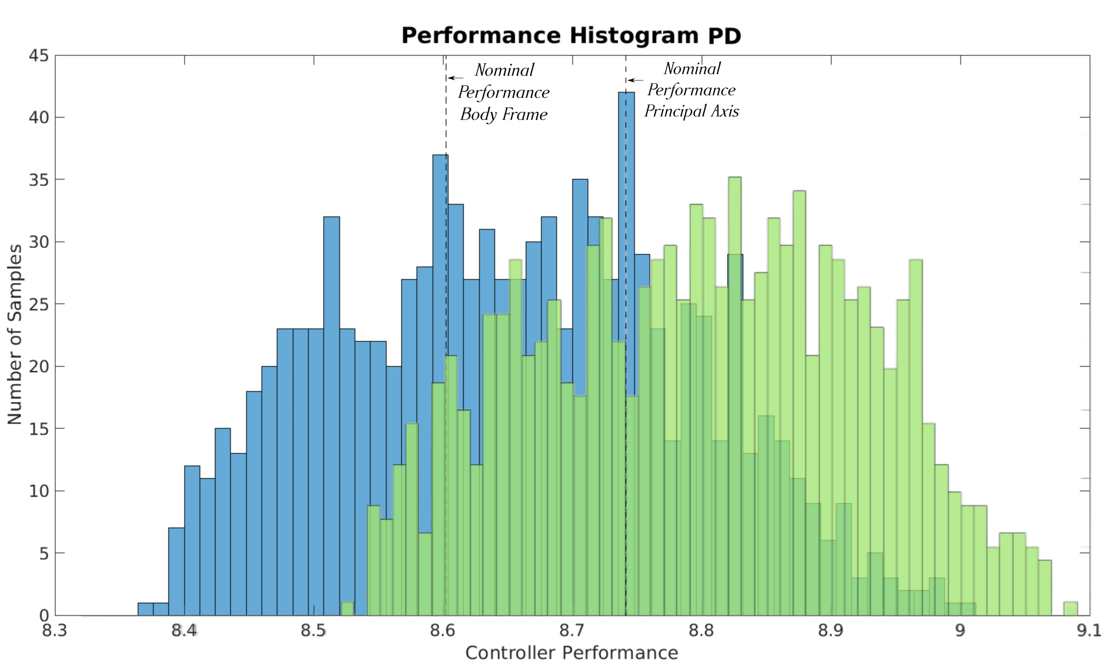


Figure 3.6: Performance Histogram of the Monte Carlo simulation for the PD controller. The green bins represent the Principal Axis and the blue bins represent the Body Frame.

Therefore, based on the obtained performance values, it is possible to conclude that the Body Frame implementation outperforms the Principal Axis. This conclusion is drawn from the observation that the range of the performance values associated with the Body Frame are lower compared to those of the Principal Axis, and its nominal performance is also lower.

3.3.2 LQR

Similar to the PD implementation, the LQR results for both Principal Axis and Body Frame implementations are very close, and are further evaluated with the help of equation 3.23.

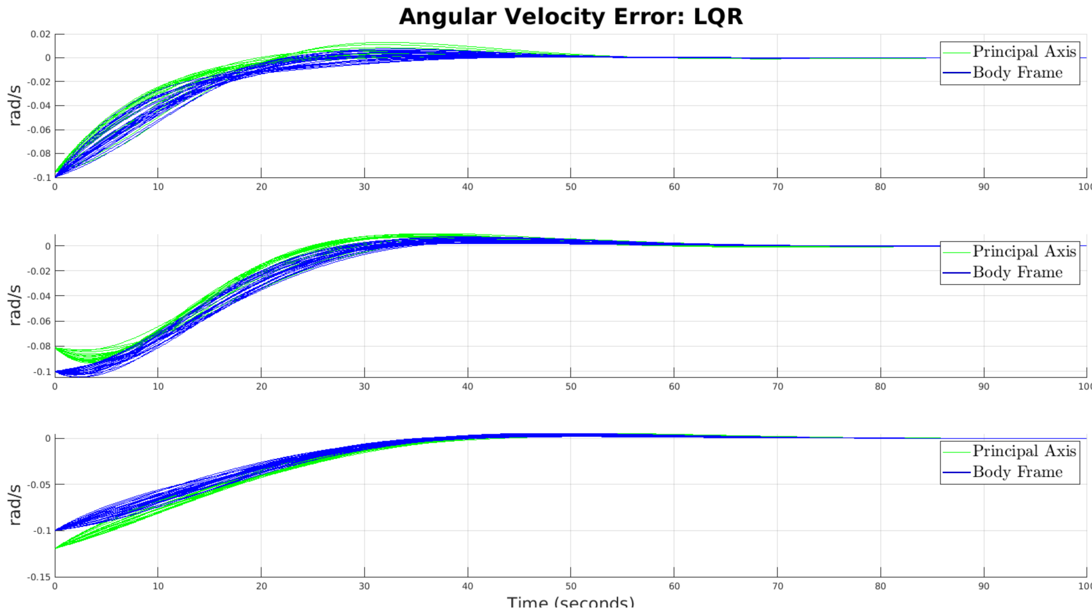


Figure 3.7: Monte Carlo simulation of the angular velocity error of the LQR controller applied in the Principal Axis and Body Frame.

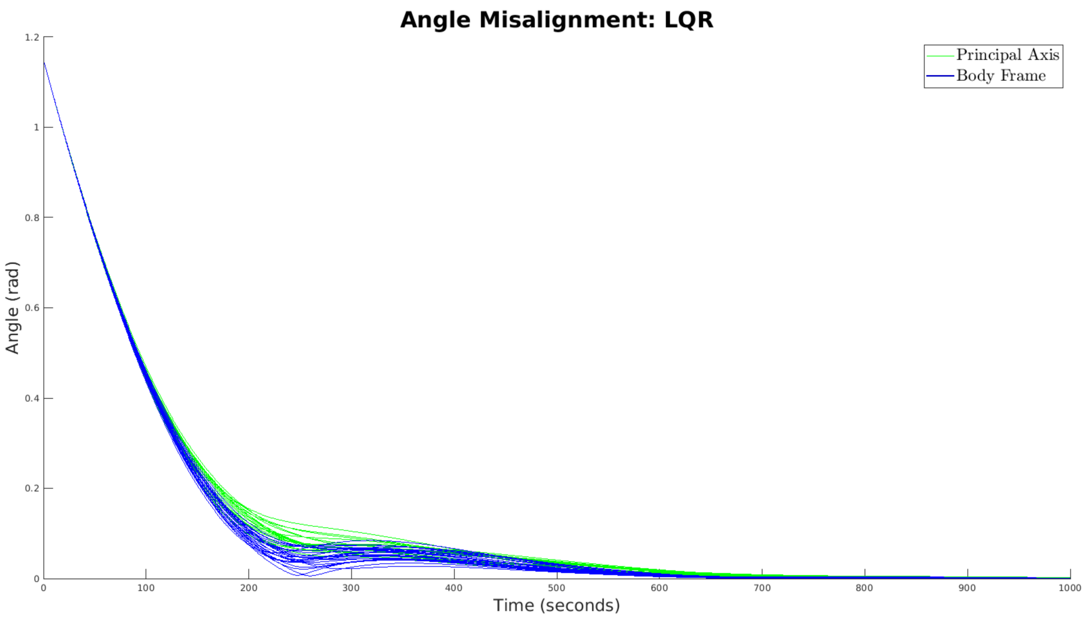


Figure 3.8: Monte Carlo simulation of the angle misalignment between the current attitude estimation and reference.

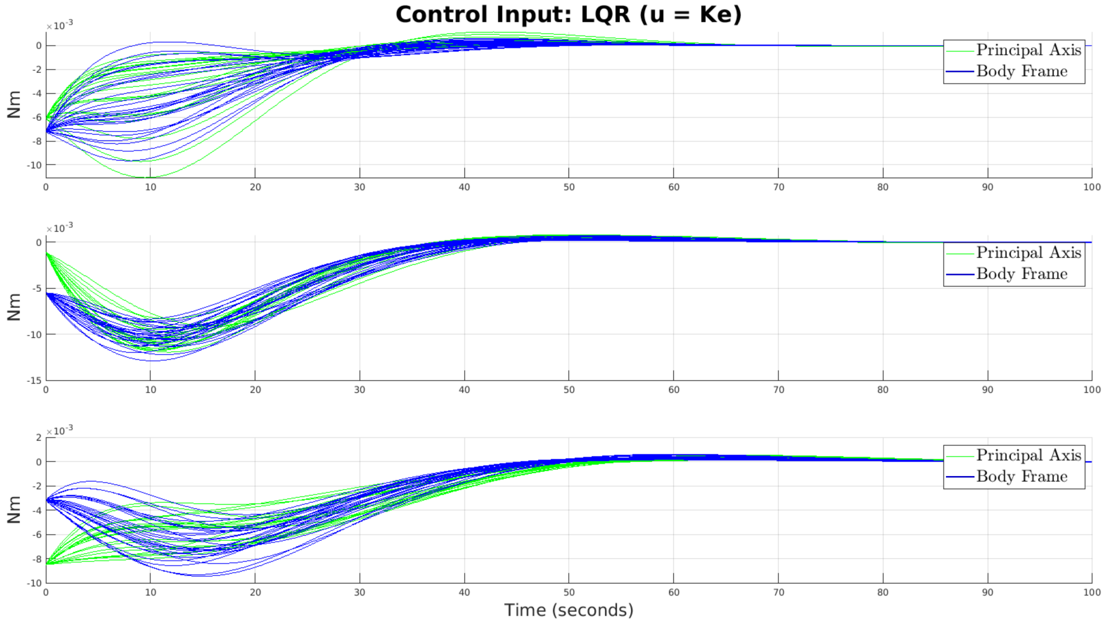


Figure 3.9: Monte Carlo simulation of the LQR control inputs.

The histogram based on 3.23 is shown below

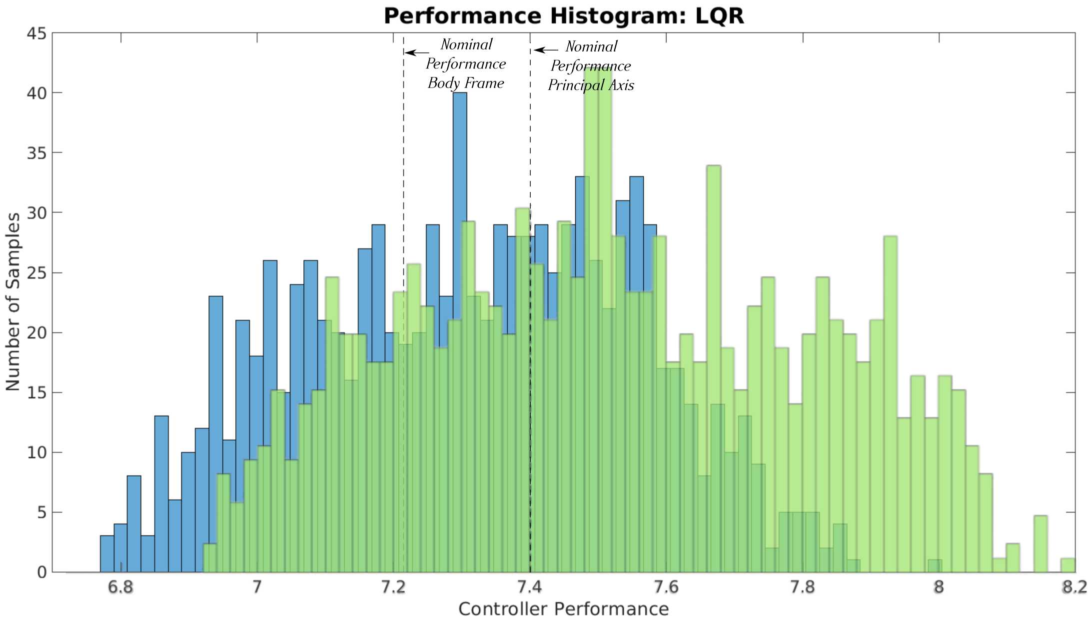


Figure 3.10: Performance Histogram of the Monte Carlo simulation for the LQR controller. The green bins represent the Principal Axis and the blue bins represent the Body Frame.

And thus is can be concluded that in the LQR case the Body Frame implementation outperforms the Principal Axis implementation.

3.3.3 Final Performance Assessment

Based on the conducted analysis, it was concluded that the performance of the Body Frame implementation, for both PD and LQR controllers, surpassed that of the Principal Axis. Therefore, as a final evaluation, a comparison is made between the results obtained from each of these performances.

Figure 3.11: Comparison between PD and LQR implemented in the Body Frame. The LQR histogram is represented in orange, while the PD histogram is represented in pink.

As a result, it can be concluded that the LQR performance is superior to the PD performance, as evidenced by the lower controller performance value reflected in its overall histogram.

Chapter 4

Advanced Attitude Control

As a step further from the previous control method using PD-control in the principal axis, the concept of adaptive control, more specifically Model Reference Adaptive Control (MRAC), is investigated. This is done with the hopes of achieving robustness towards matched uncertainties. In the following chapter, an introduction to and the motivation for the use of adaptive control is presented. Additionally, an approximation of the uncertainties utilizing basis functions is attempted. Lastly, rounding off the chapter with an implementation of the adaptive controller followed by a series of tests to assess its performance.

4.1 Model Reference Adaptive Control (MRAC)

Model Reference Adaptive Control, is as the name might suggest, a controller that allows the controlled system to "mimic" a separate reference model through adaptation. This reference model represents the desired behavior of the controlled system, given a bounded reference signal [7]. Thus, the goal of the controller is to minimize the output error between the actual system and the reference model, such that the desired behavior is maintained at all times.

As of now, two methods exist: the *Direct MRAC* (DMRAC) and the *Indirect MRAC* (IMRAC) [7]. The direct MRAC approach is called “direct” due to the fact that the gains of the control law are directly adapted, and thus imposing the desired performance on the controlled system. However, a less direct approach, hence the name “indirect” MRAC, is to first adapt/estimate the unknown system parameters, typically the $\boldsymbol{A}$ and $\boldsymbol{B}$ matrices of its state-space representation, and then use those estimates to calculate the control gains based on some matching conditions [7]. More on the matching conditions will be presented later. From these two methods, the DMRAC is chosen for the implementation.

4.1.1 The Direct MRAC Structure

The closed-loop system structure of the DMRAC controller can be seen illustrated in the block diagram in Figure 4.1. Fundamentally, as with all control systems, it is composed of the System, and the Controller. In addition to the fundamentals, the aforementioned Reference Model having the desired dynamics. Completing the control structure is an Adaptive Law block comprised of the functions that tweak the gains and weights of the control law at every time step. Consequently ensuring that the plant output tracks that of the reference model. This adaptation is indicated by the diagonal line passing through the controller block.

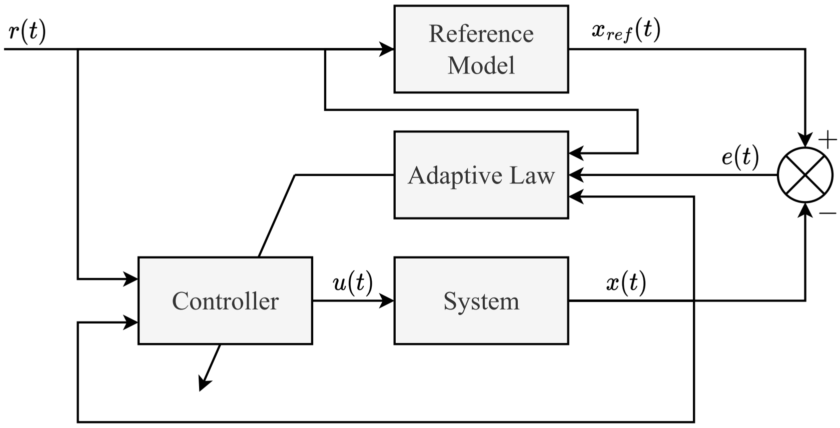


Figure 4.1: DMRAC closed-loop system block diagram. The diagonal arrow passing through the controller block indicates that the adaptive laws are tweaking the variables of the controller, without directly being fed into the controller itself.

Seen in Figure 4.1 are the different signals depicted. Firstly is $r(t) \in \mathbb{R}^m$ which is the reference signal, secondly, $u(t) \in \mathbb{R}^m$ the system control input, and lastly the model-tracking error given by $e(t) = x(t) - x_{ref}(t)$, calculated using the system states $x(t) \in \mathbb{R}^n$, and reference model states, $x_{ref}(t) \in \mathbb{R}^n$.

4.1.2 Control Design

The nonlinear multiple-input multiple-output (MIMO) systems to undergo MRAC control are expressed in the linear form [7]

$$\dot{x} = \boldsymbol{A}x + \boldsymbol{B\Lambda}(u + f(x)) \tag{4.1}$$

which is a slight modification of the well-known open-loop state-space representation. Restated, is the system state vector $x \in \mathbb{R}^n$, the control input $u \in \mathbb{R}^m$, the system matrix $\boldsymbol{A} \in \mathbb{R}^{n\times n}$, in this case unknown, and the known input matrix $\boldsymbol{B} \in \mathbb{R}^{n\times m}$. In addition, an unknown matrix $\Lambda \in \mathbb{R}^{m\times m}$ assumed diagonal with strictly positive elements λ_i and denoting the control effectiveness uncertainty. Lastly, the linear/nonlinear vector-function $f(x)$, mapping from $\mathbb{R}^n$ to $\mathbb{R}^m$ representing the matched uncertainty in the system [7]. The vector-function $f(x)$ consists of N (unknown or known) state-dependent basis functions $\Phi(x) = [\,\varphi_1(x) \;\ldots\; \varphi_N(x)\,]^T$ that each are locally Lipschitz-continuous, and has unknown weights $\boldsymbol{\Theta} \in \mathbb{R}^{N\times m}$ [7]. Together they form the matched uncertainty term:

$$f(x) = \boldsymbol{\Theta}^T \Phi(x) \tag{4.2}$$

Substituting into (4.1) gives

$$\dot{x} = \boldsymbol{A}x + \boldsymbol{B\Lambda}(u + \boldsymbol{\Theta}^T \Phi(x)) \tag{4.3}$$

Not explicit to the form (4.3), but important to note however, embedded in $\boldsymbol{\Theta}^T$ are the inverted control effectiveness uncertainties, namely $\boldsymbol{\Lambda}^{-1}$, such that when multiplied by $\boldsymbol{B\Lambda}$ the control effectiveness uncertainties are cancelled [7].

Now, the goal as mentioned previously is to follow a reference model of the designers choice with same input/output dimensionality as the actual system. Such reference model can be expressed as

$$\dot{x}_{ref} = \boldsymbol{A}_{ref} x_{ref} + \boldsymbol{B}_{ref} r(t) \tag{4.4}$$

with $\mathbf{A}_{ref}$ being Hurwitz, namely that $\mathrm{Re}[\,\mathrm{eig}(\mathbf{A}_{ref})\,] < 0$, and $r(t)$ being the reference signal [7]. Thus to achieve global asymptotic convergence of the state error $e(t)$, mentioned in Subsection 4.1.1, a control law u is designed such that

$$\lim_{t\to\infty} \|e(t)\| = 0 \tag{4.5}$$

where the term global indicates convergence of the error given *any* initial condition of the states. As a first attempt on a control law to achieve the global asymptotic convergence, a typical ideal control law of DMRAC is presented

$$u = \boldsymbol{K}_x x + \boldsymbol{K}_r r - \boldsymbol{\Theta}^T \Phi(x) \tag{4.6}$$

Yielding the closed-loop system

$$\dot{x} = (\boldsymbol{A} + \boldsymbol{B\Lambda K}_x)x + \boldsymbol{B\Lambda K}_r r \tag{4.7}$$

With $\boldsymbol{K}_x$ denoting the ideal state-feedback gain, and $\boldsymbol{K}_r$ the ideal reference signal feed-forward gain. Both gain matrices being fixed and unknown [7]. Since the closed-loop system in (4.7) is to track the reference model (4.4), comparing both equations give the following matching conditions

$$\begin{aligned} \boldsymbol{A} + \boldsymbol{B\Lambda K}_x &= \boldsymbol{A}_{ref} \\ \boldsymbol{B\Lambda K}_r &= \boldsymbol{B}_{ref} \end{aligned} \tag{4.8}$$

If ideal gains $\boldsymbol{K}_x$ and $\boldsymbol{K}_r$ exist and satisfy (4.8), then the closed-loop system (4.7) undoubtedly performs exactly like the reference model. However, in reality, the existence of these ideal fixed gains that satisfy the matching conditions (4.8) is not guaranteed. Although, if the structure of $\boldsymbol{A}$ is known, as it usually is in practice, one can design $\boldsymbol{A}_{ref}$ and $\boldsymbol{B}_{ref}$ accordingly such that at least one pair of ideal gain matrices exists [7]. Here, the term 'structure' is understood as possibly a linearization of the nonlinear dynamics around some operating point, as done in subsection 3.1.2. Since the ideal gains are not guaranteed a different approach can be taken, namely to estimate them. Assuming however, that the ideal $\boldsymbol{K}_x$ and $\boldsymbol{K}_r$ exists, one can propose the following control law

$$u = \hat{\boldsymbol{K}}_x x + \hat{\boldsymbol{K}}_r r - \hat{\boldsymbol{\Theta}}^T \Phi(x) \tag{4.9}$$

where the hat denotes the estimate of the ideal matrices in (4.6). Through *inverse Lyapunov analysis*, described in [7], these quantities can be computed on-line. Utilizing the above mentioned control law, the closed-loop dynamics then become

$$\dot{x} = (\boldsymbol{A} + \boldsymbol{B\Lambda}\hat{\boldsymbol{K}}_x)x + \boldsymbol{B\Lambda}\big[\hat{\boldsymbol{K}}_r r - (\hat{\boldsymbol{\Theta}} - \boldsymbol{\Theta})^T \Phi(x)\big] \tag{4.10}$$

and the closed-loop tracking error dynamics

$$\begin{aligned} \dot{e} &= \dot{x} - \dot{x}_{\mathit{ref}} \\ &= (\boldsymbol{A} + \boldsymbol{B}\Lambda\hat{\boldsymbol{K}}_x)x + \boldsymbol{B\Lambda}\big[\hat{\boldsymbol{K}}_r r - (\hat{\boldsymbol{\Theta}} - \boldsymbol{\Theta})^T \Phi(x)\big] - (\boldsymbol{A}_{\mathit{ref}}\, x_{\mathit{ref}} + \boldsymbol{B}_{\mathit{ref}}\, r) \end{aligned} \tag{4.11}$$

Additionally, by using the matching conditions in (4.8), the error dynamics can be further reduced

$$\begin{aligned} \dot{e} &= \big(\underbrace{\boldsymbol{A}_{\mathit{ref}} - \boldsymbol{B\Lambda K}_x}_{\boldsymbol{A}} + \boldsymbol{B\Lambda}\hat{\boldsymbol{K}}_x\big)x + \boldsymbol{B\Lambda}\big[\hat{\boldsymbol{K}}_r r - \big(\hat{\boldsymbol{\Theta}} - \boldsymbol{\Theta}\big)^T \Phi(x)\big] - (\boldsymbol{A}_{\mathit{ref}}\, x_{\mathit{ref}} + \underbrace{\boldsymbol{B\Lambda K}_r}_{\boldsymbol{B}_{\mathit{ref}}} r) \\ &= \boldsymbol{A}_{\mathit{ref}}\, e + \boldsymbol{B\Lambda}\underbrace{\big(\hat{\boldsymbol{K}}_x - \boldsymbol{K}_x\big)}_{\boldsymbol{\Delta K}_x} x + \boldsymbol{B\Lambda}\underbrace{\big(\hat{\boldsymbol{K}}_r - \boldsymbol{K}_r\big)}_{\boldsymbol{\Delta K}_r} r - \boldsymbol{B\Lambda}\underbrace{\big(\hat{\boldsymbol{\Theta}} - \boldsymbol{\Theta}\big)^T}_{\boldsymbol{\Delta\Theta}^T} \Phi(x) \\ &= \boldsymbol{A}_{\mathit{ref}}\, e + \boldsymbol{B\Lambda}\big[\boldsymbol{\Delta K}_x x + \boldsymbol{\Delta K}_r r - \boldsymbol{\Delta\Theta}^T \Phi(x)\big] \end{aligned} \tag{4.12}$$

Yielding the fully derived tracking error dynamics with the delta ($\boldsymbol{\Delta}$) terms signifying parameter estimation errors [7]. Next, some stability analysis of the error dynamics (4.12) using Lyapunov's stability theorem is performed.

Deriving the Adaptive Laws: Lyapunov method

Lyapunov's stability theorem is widely used to analyze a controlled system's ability to achieve closed-loop stability. In the case of DMRAC control the Lyapunov stability theorem, shall assist in deriving the adaptive laws that allow for the on-line adaptation/estimation of the parameter matrices $\hat{\boldsymbol{K}}_x$, $\hat{\boldsymbol{K}}_r$ and $\hat{\boldsymbol{\Theta}}$ in (4.9). Generally, parameter convergence in (D)MRAC is NOT guaranteed unless the reference signal satisfies, what is known as, the persistency of excitation (PE) conditions [7]. These conditions however are difficult to verify numerically and are therefore left out of the scope of this thesis.

Nevertheless, even if the parameters fail to converge to their ideal values, the importance lies in ensuring the limit in (4.5), since convergence of the tracking error implies perfect tracking of the reference model. To begin the stability analysis of the error dynamics, Lyapunov's stability theorem is presented in Theorem 4.1.1 below.

Theorem 4.1.1 (Lyapunov's stability theorem [5]) *Let $f(x)$ be a locally Lipschitz function defined over a domain $D \subset \mathbb{R}^n$, which contains the origin, and $f(0) = 0$. Let $V(x)$ be a* *continuously differentiable function defined over D such that*

$$V(0) = 0 \quad \text{and} \quad V(x) > 0, \quad \forall x \neq 0 \in D \tag{4.13}$$

$$\dot{V}(x) \leq 0, \quad \forall x \in D \tag{4.14}$$

Then, the origin is a stable equilibrium point of $\dot{x} = f(x)$. Moreover, if

$$\dot{V}(x) < 0, \quad \forall x \neq 0 \in D \tag{4.15}$$

then the origin is asymptotically stable. Furthermore, if $D = \mathbb{R}^n$, (4.13) *and* (4.15) *hold for all $x \neq 0$, and*

$$\|x\| \to \infty \quad \Rightarrow \quad V(x) \to \infty \tag{4.16}$$

then the origin is globally asymptotically stable [5].

Summarizing the theorem, a Lyapunov candidate function $V(x)$ has to be strictly positive, and zero in zero in order to be a candidate function. The stability is analyzed using the time derivative, and based on its negative definiteness, it is either stable (4.14), asymptotically stable (4.15) or globally asymptotically stable if the additional condition of radially unboundedness (4.16) is satisfied.

To begin the analysis [7] suggests a radially unbounded Lyapunov candidate function of the form

$$V(\psi) = e^T \boldsymbol{P} e + \operatorname{tr}\left(\left[\boldsymbol{\Delta K}_x^T \boldsymbol{\Gamma}_x^{-1} \boldsymbol{\Delta K}_x + \boldsymbol{\Delta K}_r^T \boldsymbol{\Gamma}_r^{-1} \boldsymbol{\Delta K}_r + \boldsymbol{\Delta \Theta}^T \boldsymbol{\Gamma}_\Theta^{-1} \boldsymbol{\Delta \Theta}\right] \boldsymbol{\Lambda}\right) \tag{4.17}$$

with

$$\psi = (e,\, \boldsymbol{\Delta K}_x,\, \boldsymbol{\Delta K}_r,\, \boldsymbol{\Delta \Theta})$$

where $\boldsymbol{P} = \boldsymbol{P}^T > 0$ is a positive symmetric matrix satisfying the algebraic Lyapunov equation

$$\boldsymbol{P A}_{ref} + \boldsymbol{A}_{ref}^T \boldsymbol{P} = -\boldsymbol{Q} \tag{4.18}$$

for arbitrary $\boldsymbol{Q} = \boldsymbol{Q}^T > 0$. Furthermore, the positive symmetric rates of adaptation are introduced, namely $\boldsymbol{\Gamma}_x$, $\boldsymbol{\Gamma}_r$, and $\boldsymbol{\Gamma}_\Theta$. Additionally, the first term within the trace regarding the state feedback, is changed to $\boldsymbol{\Delta K}_x \boldsymbol{\Gamma}_x^{-1} \boldsymbol{\Delta K}_x^T$, to account for the dimensionality mismatch with the multiplication of $\boldsymbol{\Lambda}$. With the Lyapunov candidate function the time derivative can then be computed as [5][7]

$$\dot{V}(\psi) = \frac{\partial V(\psi)}{\partial \psi} \frac{d\psi}{dt} \tag{4.19}$$

Yielding

$$\dot{V}(\psi) = \dot{e}^T \boldsymbol{P} e + e^T \boldsymbol{P} \dot{e} + 2 \operatorname{tr}\left(\left[\boldsymbol{\Delta K}_x \boldsymbol{\Gamma}_x^{-1} \dot{\hat{\boldsymbol{K}}}_x^T + \boldsymbol{\Delta K}_r^T \boldsymbol{\Gamma}_r^{-1} \dot{\hat{\boldsymbol{K}}}_r + \boldsymbol{\Delta \Theta}^T \boldsymbol{\Gamma}_\Theta^{-1} \dot{\hat{\boldsymbol{\Theta}}}\right] \boldsymbol{\Lambda}\right) \tag{4.20}$$

Inserting the error dynamics in (4.12) further gives

$$\begin{aligned}\dot{V}(\psi) &= (\boldsymbol{A}_{\mathit{ref}}\, e + \boldsymbol{B\Lambda}\big[\boldsymbol{\Delta K}_x x + \boldsymbol{\Delta K}_r r - \boldsymbol{\Delta\Theta}^T\Phi(x)\big])^T \boldsymbol{P} e \\ &\quad + e^T \boldsymbol{P}(\boldsymbol{A}_{\mathit{ref}}\, e + \boldsymbol{B\Lambda}\big[\boldsymbol{\Delta K}_x x + \boldsymbol{\Delta K}_r r - \boldsymbol{\Delta\Theta}^T\Phi(x)\big]) \\ &\quad + 2\,\mathrm{tr}\left(\left[\boldsymbol{\Delta K}_x \boldsymbol{\Gamma}_x^{-1}\dot{\hat{\boldsymbol{K}}}_x^T + \boldsymbol{\Delta K}_r^T \boldsymbol{\Gamma}_r^{-1}\dot{\hat{\boldsymbol{K}}}_r + \boldsymbol{\Delta\Theta}^T \boldsymbol{\Gamma}_\Theta^{-1}\dot{\hat{\boldsymbol{\Theta}}}\right]\boldsymbol{\Lambda}\right)\end{aligned} \tag{4.21}$$

Collecting the terms

$$\begin{aligned}\dot{V}(\psi) &= e^T(\boldsymbol{A}_{\mathit{ref}}^T \boldsymbol{P} + \boldsymbol{P A}_{\mathit{ref}})e + 2e^T \boldsymbol{PB\Lambda}\big[\boldsymbol{\Delta K}_x x + \boldsymbol{\Delta K}_r r - \boldsymbol{\Delta\Theta}^T\Phi(x)\big] \\ &\quad + 2\,\mathrm{tr}\left(\left[\boldsymbol{\Delta K}_x \boldsymbol{\Gamma}_x^{-1}\dot{\hat{\boldsymbol{K}}}_x^T + \boldsymbol{\Delta K}_r^T \boldsymbol{\Gamma}_r^{-1}\dot{\hat{\boldsymbol{K}}}_r + \boldsymbol{\Delta\Theta}^T \boldsymbol{\Gamma}_\Theta^{-1}\dot{\hat{\boldsymbol{\Theta}}}\right]\boldsymbol{\Lambda}\right)\end{aligned} \tag{4.22}$$

Noting that the first term of (4.22) is the Lyapunov equation (4.18). Using this and reducing further, yields:

$$\begin{aligned}\dot{V}(\psi) = -e^T \boldsymbol{Q} e &+ \left[2e^T \boldsymbol{PB\Lambda\Delta K}_x x + 2\,\mathrm{tr}\left(\boldsymbol{\Delta K}_x \boldsymbol{\Gamma}_x^{-1}\dot{\hat{\boldsymbol{K}}}_x^T\,\boldsymbol{\Lambda}\right)\right] \\ &+ \left[2e^T \boldsymbol{PB\Lambda\Delta K}_r r + 2\,\mathrm{tr}\left(\boldsymbol{\Delta K}_r^T \boldsymbol{\Gamma}_r^{-1}\dot{\hat{\boldsymbol{K}}}_r\,\boldsymbol{\Lambda}\right)\right] \\ &+ \left[2e^T \boldsymbol{PB\Lambda\Delta\Theta}\Phi(x) + 2\,\mathrm{tr}\left(\boldsymbol{\Delta\Theta}^T \boldsymbol{\Gamma}_\Theta^{-1}\dot{\hat{\boldsymbol{\Theta}}}\,\boldsymbol{\Lambda}\right)\right]\end{aligned} \tag{4.23}$$

Now using the following vector trace identity [7]

$$a^T b = \mathrm{tr}(b a^T) \tag{4.24}$$

the first terms in the square brackets in (4.23) can be rewritten as

$$\begin{aligned}\underbrace{e^T \boldsymbol{PB\Lambda}}_{a^T}\underbrace{\boldsymbol{\Delta K}_x x}_{b} &= \mathrm{tr}\left(\underbrace{\boldsymbol{\Delta K}_x x}_{b}\underbrace{e^T \boldsymbol{PB\Lambda}}_{a^T}\right) \\ \underbrace{e^T \boldsymbol{PB\Lambda}}_{a^T}\underbrace{\boldsymbol{\Delta K}_r^T r}_{b} &= \mathrm{tr}\left(\underbrace{\boldsymbol{\Delta K}_r^T r}_{b}\underbrace{e^T \boldsymbol{PB\Lambda}}_{a^T}\right) \\ \underbrace{e^T \boldsymbol{PB\Lambda}}_{a^T}\underbrace{\boldsymbol{\Delta\Theta}^T\Phi(x)}_{b} &= \mathrm{tr}\left(\underbrace{\boldsymbol{\Delta\Theta}^T\Phi(x)}_{b}\underbrace{e^T \boldsymbol{PB\Lambda}}_{a^T}\right)\end{aligned} \tag{4.25}$$

substituting (4.25) into (4.23) then yields

$$\begin{aligned}\dot{V}(\psi) = -e^T \boldsymbol{Q} e &+ 2\,\mathrm{tr}\left(\boldsymbol{\Delta K}_x\left[\boldsymbol{\Gamma}_x^{-1}\dot{\hat{\boldsymbol{K}}}_x^T + x e^T \boldsymbol{PB}\right]\boldsymbol{\Lambda}\right) \\ &+ 2\,\mathrm{tr}\left(\boldsymbol{\Delta K}_r^T\left[\boldsymbol{\Gamma}_r^{-1}\dot{\hat{\boldsymbol{K}}}_r + r e^T \boldsymbol{PB}\right]\boldsymbol{\Lambda}\right) \\ &+ 2\,\mathrm{tr}\left(\boldsymbol{\Delta\Theta}^T\left[\boldsymbol{\Gamma}_\Theta^{-1}\dot{\hat{\boldsymbol{\Theta}}} + \Phi(x) e^T \boldsymbol{PB}\right]\boldsymbol{\Lambda}\right)\end{aligned} \tag{4.26}$$

From here, it is easy to verify, that choosing the following adaptive laws

$$\begin{aligned}\dot{\hat{\boldsymbol{K}}}_x &= -\left(\boldsymbol{\Gamma}_x x e^T \boldsymbol{P}\boldsymbol{B}\right)^T \\ \dot{\hat{\boldsymbol{K}}}_r &= -\boldsymbol{\Gamma}_r r(t) e^T \boldsymbol{P}\boldsymbol{B} \\ \dot{\hat{\boldsymbol{\Theta}}} &= \boldsymbol{\Gamma}_\Theta \Phi(x) e^T \boldsymbol{P}\boldsymbol{B}\end{aligned} \tag{4.27}$$

will render the derivative of the Lyapunov candidate function $V(\psi)$, negative semi-definite:

$$\dot{V}(\psi) = -e^T \boldsymbol{Q} e \leq 0 \tag{4.28}$$

This however, only makes the closed-loop error dynamics stable and not asymptotically stable according to Theorem 4.1.1. Although through what is known as Barbalat's Lemma [7], that requires the Lyapunov candidate function to satisfy certain conditions, one achieves $\lim_{t\to\infty} \dot{V}(\psi) = 0$ and thus $\lim_{t\to\infty} \|e(t)\| = 0$. Hence, *global asymptotic* convergence of the tracking error is guaranteed. The three conditions on $V(\psi)$ of Barbalat's lemma are [7]:

1. $V(\psi)$ is lower bounded
2. $\dot{V}(\psi) \leq 0$, derivative is negative semi-definite
3. $\dot{V}(\psi)$ is uniform continuous, i.e, $\ddot{V}(\psi)$ is bounded

The condition of the first item is upheld as $V(\psi)$ is a quadratic function that is radially unbounded and thus has a lower bound at its base, namely the origin. Mathematically this would be written as $|V(\psi)| \geq 0$. The second condition is already obtained in (4.28), and lastly, the third condition, namely checking the boundedness of its double derivative. By using (4.19) the double derivative of the Lyapunov candidate function becomes

$$\ddot{V}(\psi) = -2e^T \boldsymbol{Q}\dot{e} \tag{4.29}$$

where it is noticed that $e(t)$ has already been established bounded as a result of the stable closed-loop error dynamics along with the parameter estimation errors $\boldsymbol{\Delta K}_x(t)$, $\boldsymbol{\Delta K}_r(t)$, $\boldsymbol{\Delta\Theta}(t)$, and the parameter estimates $\hat{\boldsymbol{K}}_x(t)$, $\hat{\boldsymbol{K}}_r(t)$, $\hat{\boldsymbol{\Theta}}(t)$. Since $e(t)$ is bounded, both $x(t)$ and $x_{ref}(t)$ must as a result also be bounded. Lastly, the reference signal $r(t)$ is assumed bounded by default for the design of the (D)MRAC scheme, and $\Phi(x)$ being locally Lipschitz functions of the states, must thus also be bounded.

Therefore, it can then be concluded that the error dynamics $\dot{e}(t)$ are bounded, thereby also making $\ddot{V}(\psi)$ bounded. Finally, since the conditions of Barbalat's lemma have been proven satisfied, *global asymptotic* stability of the tracking error is achieved. Next, the controller implementation is done.

4.1.3 Controller Implementation

With the proof of tracking error convergence shown using the control law in (4.9) and its respective adaptive laws in (4.27), the implementation of the controller is carried out in the following. As for the parameters, the matrices $\boldsymbol{A}$ and $\boldsymbol{B}$ remain unchanged from the linearization in Subsection 3.1.2, the control effectiveness uncertainty is chosen to be $\boldsymbol{\Lambda} = 0.75\boldsymbol{I}_3$.

The desired closed-loop performance of the DMRAC controlled system, namely that of the reference model system, is obtained by computing the state feedback gain $\boldsymbol{K}_x$ and feedforward gain $\boldsymbol{K}_r$ followed by computing $\boldsymbol{A}_{ref}$ and $\boldsymbol{B}_{ref}$ using the model matching conditions in (4.8). Through LQR, $\boldsymbol{K}_x$ is acquired. Choosing the matrices

$$\boldsymbol{Q} = 10\,\boldsymbol{I}_6 \qquad \wedge \qquad \boldsymbol{H} = 5 \cdot 0.1^{-2}\,\boldsymbol{I}_3 \tag{4.30}$$

yields the state feedback gain

$$\boldsymbol{K}_x = \begin{bmatrix} -0.1414 & 0 & 0 & -0.4699 & -0.0012 & -0.0019 \\ 0 & -0.1414 & 0 & -0.0012 & -0.5144 & -0.0080 \\ 0 & 0 & -0.1414 & -0.0019 & -0.0080 & -0.5541 \end{bmatrix} \tag{4.31}$$

For the feedforward gain $\boldsymbol{K}_r$, a quick steady-state analysis is done. In steady state $\dot{x} = 0$, thus a closed-loop system with state feedback and feedforward can be written as

$$\begin{aligned} 0 &= (\boldsymbol{A} + \boldsymbol{B}\boldsymbol{\Lambda}\boldsymbol{K}_x)x_{ss} + \boldsymbol{B}\boldsymbol{\Lambda}\boldsymbol{K}_r r(t) \\ y_{ss} &= \boldsymbol{C}x_{ss} \end{aligned} \tag{4.32}$$

isolating the state vector

$$x_{ss} = -(\boldsymbol{A} + \boldsymbol{B}\boldsymbol{\Lambda}\boldsymbol{K}_x)^{-1}\boldsymbol{B}\boldsymbol{\Lambda}\boldsymbol{K}_r r(t) \tag{4.33}$$

yielding the output

$$y_{ss} = -\boldsymbol{C}(\boldsymbol{A} + \boldsymbol{B}\boldsymbol{\Lambda}\boldsymbol{K}_x)^{-1}\boldsymbol{B}\boldsymbol{\Lambda}\boldsymbol{K}_r r(t) \tag{4.34}$$

As can be seen in the steady-state output, that for the output to follow a step reference, the equality $-\boldsymbol{C}(\boldsymbol{A} + \boldsymbol{B}\boldsymbol{\Lambda}\boldsymbol{K}_x)^{-1}\boldsymbol{B}\boldsymbol{\Lambda}\boldsymbol{K}_r = \boldsymbol{I}$ inevitably needs to be satisfied. However, since $r(t) \in \mathbb{R}^3$ and $\boldsymbol{C} \in \mathbb{R}^{6\times 6}$, the $\boldsymbol{C}$-matrix has to be reduced for the dimensions to match. Hence $\boldsymbol{C} = [\boldsymbol{I}_3\ \boldsymbol{0}_3]$ is chosen, thus giving the following feedforward gain

$$\boldsymbol{K}_r = -(\boldsymbol{C}(\boldsymbol{A} + \boldsymbol{B}\boldsymbol{\Lambda}\boldsymbol{K}_x)^{-1}\boldsymbol{B}\boldsymbol{\Lambda})^{-1} = 0.1414\,\boldsymbol{I}_3 \tag{4.35}$$

or simply

$$\boldsymbol{K}_r = -\boldsymbol{K}_x(1{:}3, 1{:}3) \tag{4.36}$$

where the notation $\boldsymbol{K}_x(1{:}3, 1{:}3)$ denotes the first three rows and columns of $\boldsymbol{K}_x$.
With the gains obtained, $\boldsymbol{A}_{ref}$ and $\boldsymbol{B}_{ref}$ can be computed and thus completing the reference

model (4.4). Its response to a step reference signal input $r(t)$ is shown below:

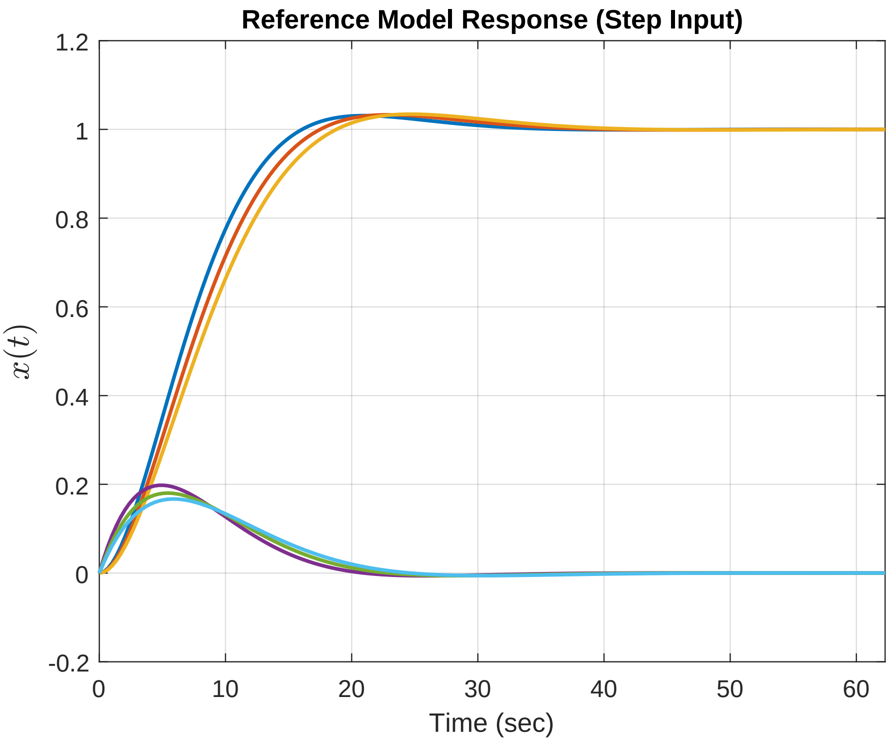


Figure 4.2: Reference model step response.

The response observed in Figure 4.2 is deemed a decent response for the DMRAC controlled system to track and hence, a first attempt on a DMRAC controller implementation is done next.

DMRAC: Attempt #1

For the first attempt, the open-loop system in (4.1) is assumed to be free of matched disturbances/uncertainties, i.e, $\Phi(x) = 0$ thus yielding the following equations for the implementation:

Open-loop system	$\dot{x} = \boldsymbol{A}x + \boldsymbol{B\Lambda}u(t)$
Reference model	$\dot{x}_{ref} = \boldsymbol{A}_{ref}x_{ref} + \boldsymbol{B}_{ref}r(t)$
Control input	$u(t) = \hat{\boldsymbol{K}}_x x + \hat{\boldsymbol{K}}_r r$
Adaptive laws	$\dot{\hat{\boldsymbol{K}}}_x = -(\boldsymbol{\Gamma}_x x e^T \boldsymbol{PB})^T$
	$\dot{\hat{\boldsymbol{K}}}_r = -\boldsymbol{\Gamma}_r r(t) e^T \boldsymbol{PB}$

Regarding the available tuning knobs, these are $\boldsymbol{Q}$, $\boldsymbol{\Gamma}_r$ and $\boldsymbol{\Gamma}_x$, and are with some trial and error chosen to be the following:

$$\boldsymbol{Q} = \boldsymbol{I}_6 \quad \wedge \quad \boldsymbol{\Gamma}_r = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 10 \end{bmatrix} \quad \wedge \quad \boldsymbol{\Gamma}_x = \begin{bmatrix} 10\,\boldsymbol{I}_3 & \boldsymbol{0}_3 \\ \boldsymbol{0}_3 & 75\,\boldsymbol{I}_3 \end{bmatrix} \tag{4.37}$$

Important to note, that $\boldsymbol{Q}$ in this case is not to be confused with that of the LQR. Here, $\boldsymbol{Q}$ is selected to allow for the solution of $\boldsymbol{P}$ in the Lyapunov equation (4.18), thus acting as an indirect tuning knob, through $\boldsymbol{P}$ in the adaptive laws. As for the implementation approach, this is represented by the pseudocode below:

Algorithm 1 DMRAC Controller

```
initialization;
  for i = 0 : dt : t_final do
  % Control signal
     u ← K̂_x x + K̂_r r − Θ̂^T Φ(x)
  % Adaptive laws
     K̂_x ← K̂_x + dt(−Γ_x x e^T PB)^T          % Backward Euler method
     K̂_r ← K̂_r + dt(−Γ_r r e^T PB)
      Θ̂ ← Θ̂ + dt(Γ_Θ Φ(x) e^T PB)
  % System and Reference model
       x ← x + dt(Ax + BΛ[u + Θ^T Φ(x)])
     x_ref ← x_ref + dt(A_ref x_ref + B_ref r)
  end for
```

The reference signal $r(t)$, will be the same series of four normalized quaternions as in the previous chapter. However, since the DMRAC open-loop system is a sixth-order state-space system, it does not include the quaternion scalar part η, thus, the reference signal will only be comprised of the vector part $\bar{q}_{1:3}$. Hence, the set of references are as follows:

$$r(t) = \left\{ \begin{bmatrix} 0 \\ 0 \\ 0 \end{bmatrix}, \begin{bmatrix} 0.8889 \\ 0.1111 \\ 0 \end{bmatrix}, \begin{bmatrix} 0.3586 \\ 0.4781 \\ 0.7171 \end{bmatrix}, \begin{bmatrix} 0.6470 \\ 0.7548 \\ 0.1078 \end{bmatrix} \right\} \tag{4.38}$$

With the proper initialization, implementing the above pseudocode in MATLAB yields the following system responses:

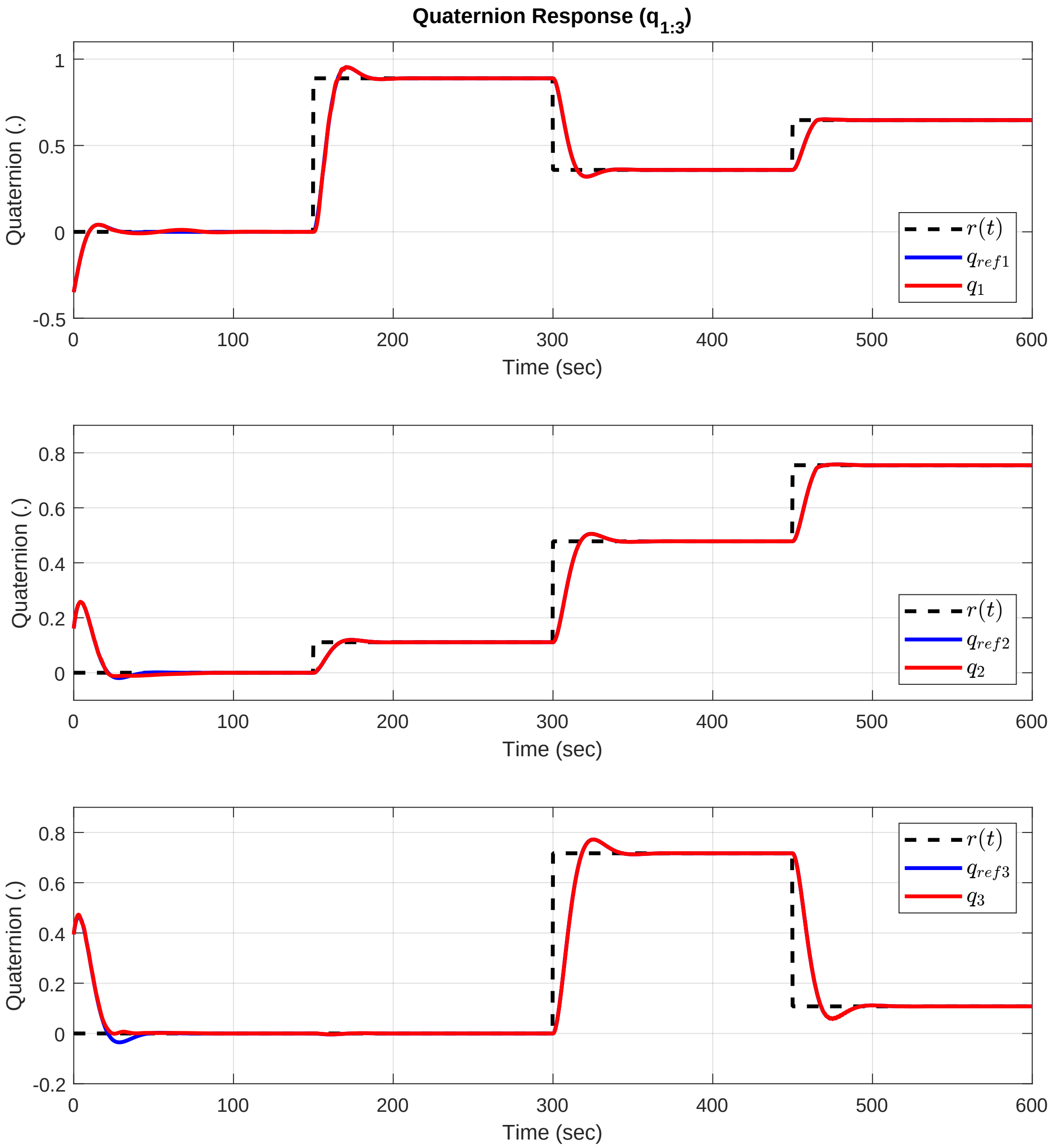


Figure 4.3: Closed-loop quaternion response without matched uncertainty and with control input saturation.

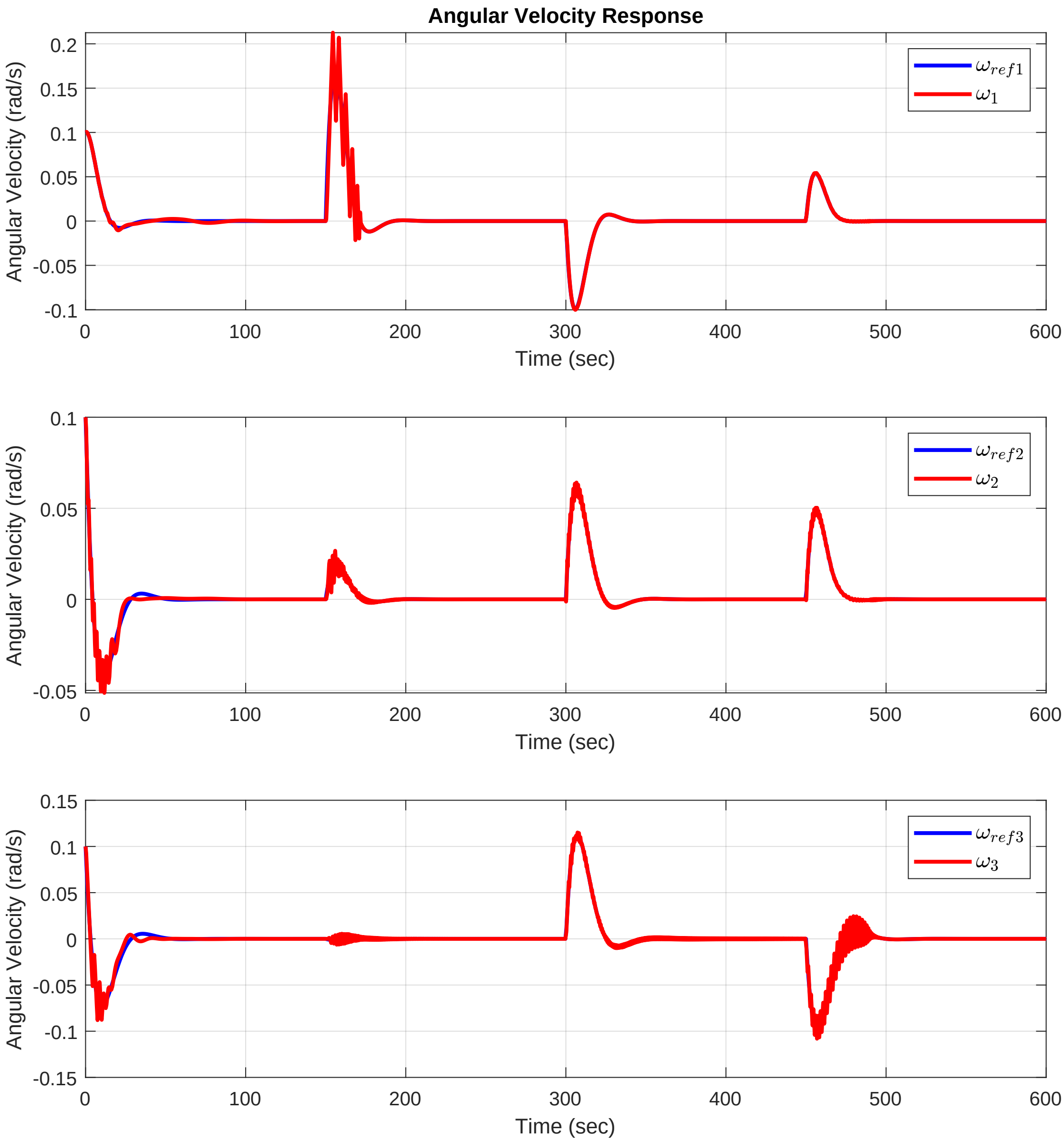


Figure 4.4: Closed-loop angular velocity response without matched uncertainty, however with control input saturation.

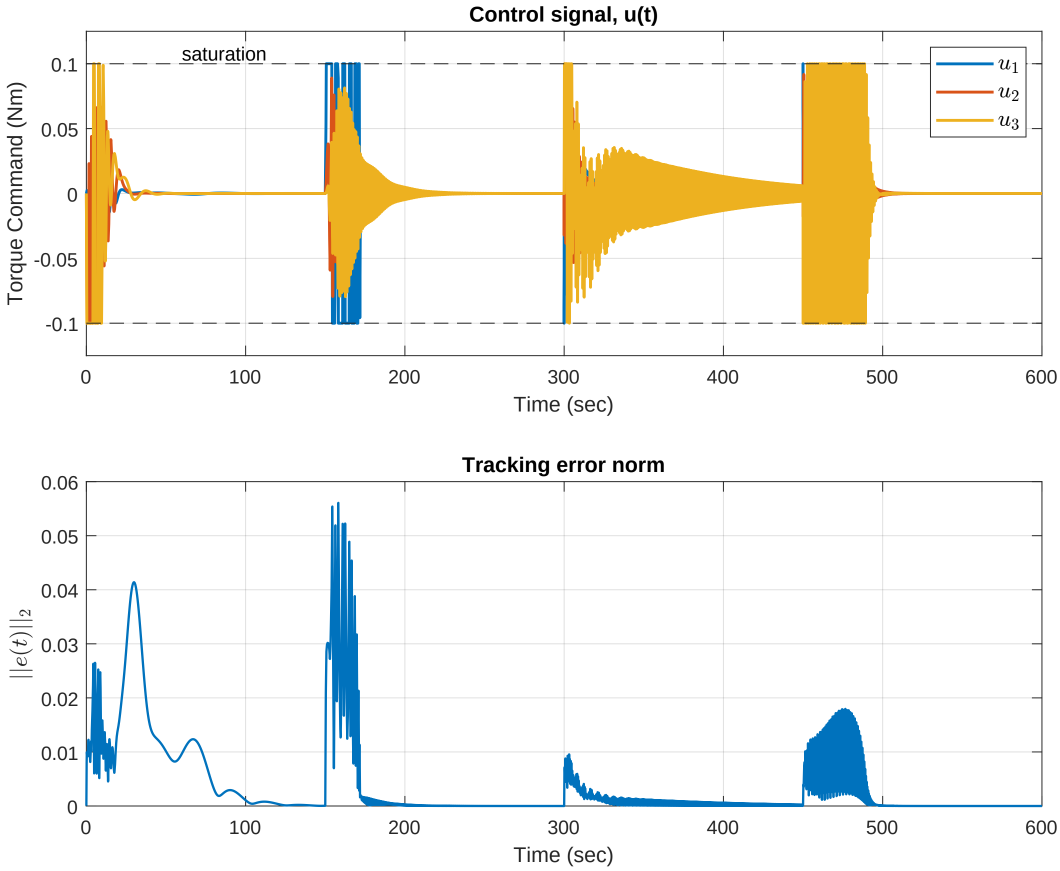


Figure 4.5: Control signal without matched uncertainty, under saturation limits. Saturation of the input can be seen.

Given the responses of Figures 4.3-4.5, the general response is very acceptable, and the tracking error norm tends to zero for each introduced step. However, there are the oscillations observed in the angular velocities during its transients. The source of the oscillation is easily located to be from the control signal, since it reaches saturation. Futhermore, despite the saturation, the high-frequency oscillations are assumed to be a side-product of the DMRAC design, as its common to see an increase in transient oscillation with higher adaptation rates [7].
To get rid of these oscillations, a simple method is introduced by [7], namely to create an observer-like addition to the reference model, modifying it to have the following dynamics:

$$\dot{x}_{ref} = \boldsymbol{A}_{ref}\, x_{ref} + \boldsymbol{B}_{ref}\, r(t) + \boldsymbol{K}_e (x - x_{ref}) \tag{4.39}$$

Consequently changing the error dynamics of (4.12) into

$$(\boldsymbol{A}_{ref} - \boldsymbol{K}_e)\, e + \boldsymbol{B\Lambda}\big[\boldsymbol{\Delta K}_x x + \boldsymbol{\Delta K}_r r - \boldsymbol{\Delta\Theta}^T \Phi(x)\big] \tag{4.40}$$

Making $\boldsymbol{K}_e$ have a direct impact on the transient response of the tracking error dynamics. This ultimately leads to a faster error convergence for any $\boldsymbol{K}_e > 0$, and therefore assists in limiting the oscillations in question [7]. A simple example is given below, choosing a single element of

$e(t)$, varying $\boldsymbol{K}_e$ and recording its impact:

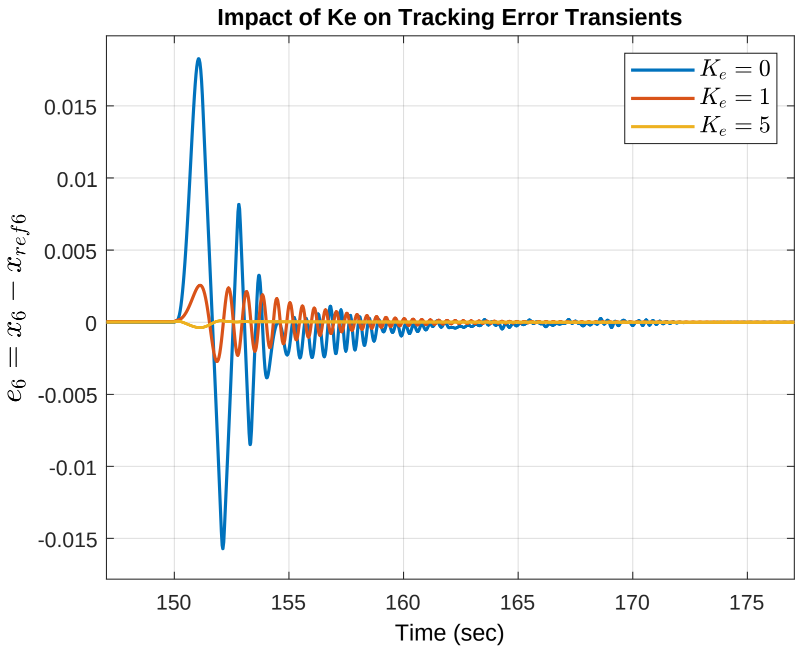


Figure 4.6: Varying $\boldsymbol{K}_e$ in order to see its impact on the transient response.

Which shows significant improvement, even with small $\boldsymbol{K}_e$, however can be subject to change for the final design. Playing around with the tracking error gain it was found that the initial response would become rather sluggish with larger values of $\boldsymbol{K}_e$, although keeping the control signal from saturating and oscillating, consequently improving the overall closed-loop system response. Nevertheless, it is assumed that the inclusion of the matched uncertainty will lead to changes in the design parameters, so the full results of using $\boldsymbol{K}_e$ are shown for the next iteration, namely with the implementation of $\Phi(x)$. In the following the choice of $\Phi(x)$ is to be determined.

Picking a Matched Uncertainty Structure

Given the adaptation laws in (4.27), the only unknown from a design standpoint, is the vector of basis functions $\Phi(x)$. The basis functions are chosen such that they resemble certain types of uncertainty or disturbance structure, if known, thereby making this structure adaptable with $\hat{\boldsymbol{\Theta}}$, and as a result cancelled with the control term $-\hat{\boldsymbol{\Theta}}^T\Phi(x)$. As for this specific case, the types of uncertainty desired to be able to handle with MRAC control is, uncertainty due to fuel consumption or fuel sloshing.

Fuel sloshing happens as a result of applying a torque in the satellite, making the fuel move in an uncontrollable manner and hence causing inaccuracies in the attitude accuracy [13]. The sloshing can be approximated by studying the behavior of both the pendulum and mass-spring system, which is done in [13], that both has oscillatory behavior. Since both systems' oscillations eventually subside due to friction, it is chosen to use a simple second-order system of the form

$$H(s) = \frac{\omega_n^2}{s^2 + 2\zeta\omega_n s + \omega_n^2} \tag{4.41}$$

which in the time domain, has the impulse response [12]

$$h(t) = \frac{\omega_n}{\sqrt{1-\zeta^2}} \mathrm{e}^{-\sigma t} \sin(\omega_d t) \tag{4.42}$$

to approximate the structure of the sloshing. Where $\sigma = \zeta\omega_n$ and $\omega_d = \omega_n\sqrt{1-\zeta^2}$ denotes the real and the imaginary part of a complex pole of the form $s = -\sigma \pm j\omega_d$, respectively. Furthermore, ζ is the damping ratio of system $H(s)$, and ω_n its (undamped) natural frequency [12]. As a first try, the "sloshing" is decided to only affect the angular velocity states, thus yielding the following basis function vector:

$$\Phi(x) = \begin{bmatrix} \boldsymbol{q}_{1:3}^T & h(t)\omega^T & 1 \end{bmatrix}^T \tag{4.43}$$

Noting now that the uncertainty structure is involved, a $\boldsymbol{\Theta}$ will have to be chosen as an "ideal" set of uncertainty coefficients to make up the matched uncertainties. Realistically, these coefficients are impossible to choose, but for simplicity chosen as

$$\boldsymbol{\Theta}^T = \begin{bmatrix} 1 & 0 & 0 & 1 & 0 & 0 & 1 \\ 0 & 1 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 1 & 0 \end{bmatrix} \tag{4.44}$$

As previously commented on equation (4.3), the coefficients are multiplied the inverse of the control effectiveness uncertainty, thus $\boldsymbol{\Theta}^T \Rightarrow \boldsymbol{\Lambda}^{-1}\boldsymbol{\Theta}^T$. Furthermore, choosing $\omega = 1.5$ and $\zeta = 0.001$ for the impulse response in (4.42), the following system matched uncertainty is obtained:

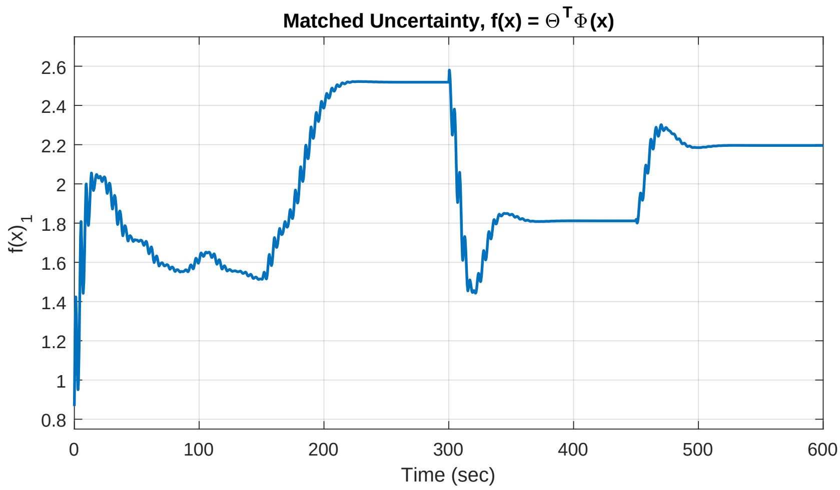


Figure 4.7: Matched uncertainty $f(x) = \mathbf{\Theta}^T\Phi(x)$. Here only the first entry of the vector function is shown.

Simulating the closed-loop system of the first implementation, now with the matched uncertainty, yields the following responses

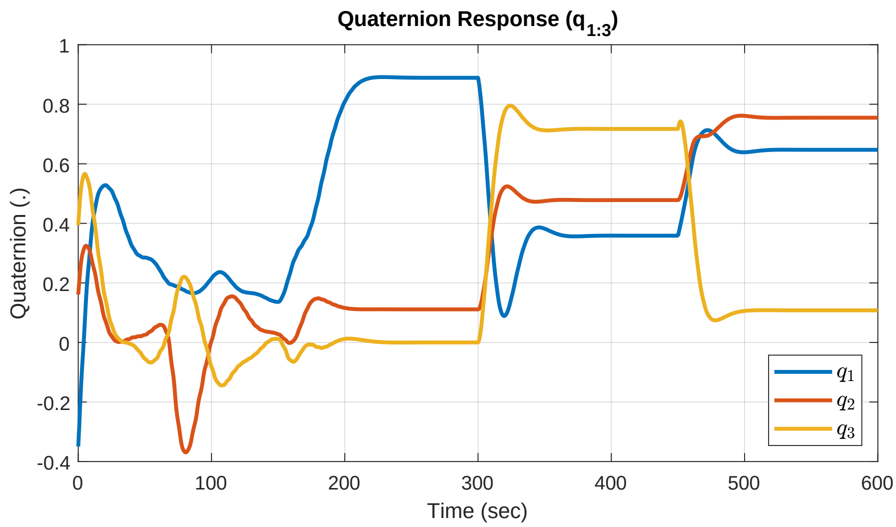


Figure 4.8: Closed-loop system quaternion response of the initial implementation, with the addition of the matched uncertainties, $f(x)$.

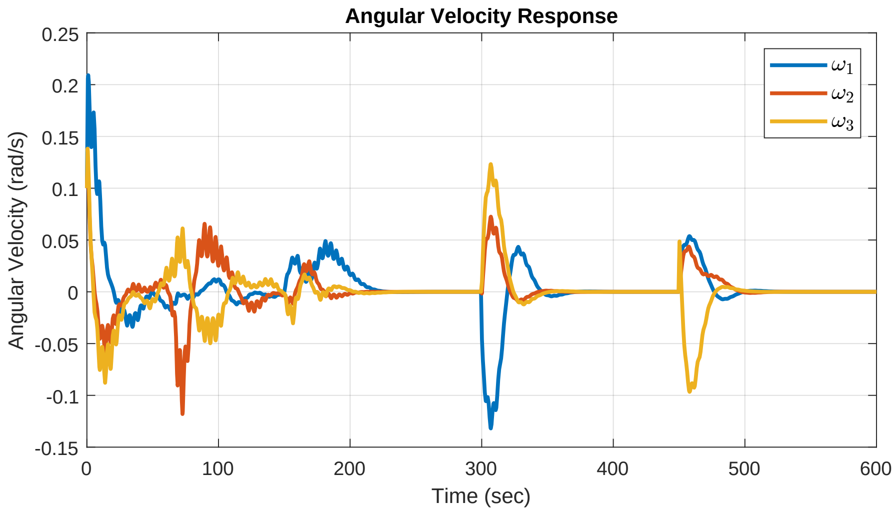


Figure 4.9: Closed-loop system angular velocity response of the initial implementation, with the addition of the matched uncertainties, $f(x)$.

Leading to the second implementation attempt, with the goal to estimate the matched uncertainty portrayed in Figure 4.7 and mitigate it through control.

DMRAC: Attempt #2

In the same manner as in the initial attempt, the equations used for the implementation are restated, with their respective alterations:

Open-loop system	$\dot{x} = \boldsymbol{A}x + \boldsymbol{B\Lambda}(u(t) + \boldsymbol{\Theta}^T\Phi(x))$
Reference model	$\dot{x}_{ref} = \boldsymbol{A}_{ref}\,x_{ref} + \boldsymbol{B}_{ref}\,r(t) + \boldsymbol{K}_e(x - x_{ref})$
Control input	$u(t) = \hat{\boldsymbol{K}}_x x + \hat{\boldsymbol{K}}_r r - \hat{\boldsymbol{\Theta}}^T\Phi(x)$
Adaptive laws	$\dot{\hat{\boldsymbol{K}}}_x = -(\boldsymbol{\Gamma}_x x e^T \boldsymbol{PB})^T$
	$\dot{\hat{\boldsymbol{K}}}_r = -\boldsymbol{\Gamma}_r r(t) e^T \boldsymbol{PB}$
	$\dot{\hat{\boldsymbol{\Theta}}} = \boldsymbol{\Gamma}_\Theta \Phi(x) e^T \boldsymbol{PB}$

The following choices of parameters for the implementation, has been made:

$$\boldsymbol{\Lambda} = 0.75\,\boldsymbol{I}_3 \quad \wedge \quad \boldsymbol{Q} = 10\,\boldsymbol{I}_6 \quad \wedge \quad \Phi(x) = \begin{bmatrix} \boldsymbol{q}_{1:3}^T & h(t)\omega^T & 1 \end{bmatrix}^T$$

For the adaptation rates $\boldsymbol{\Gamma}_x$ and $\boldsymbol{\Gamma}_r$ remain unchanged, and the uncertainty adaption rate $\boldsymbol{\Gamma}_\Theta$, along with the tracking error gain $\boldsymbol{K}_e$ are chosen as

$$\boldsymbol{\Gamma}_\Theta = 100\,\boldsymbol{I}_7 \quad \wedge \quad \boldsymbol{K}_e = 80 \tag{4.45}$$

Disregarding the saturation on the input for now, the resulting responses are then obtained:

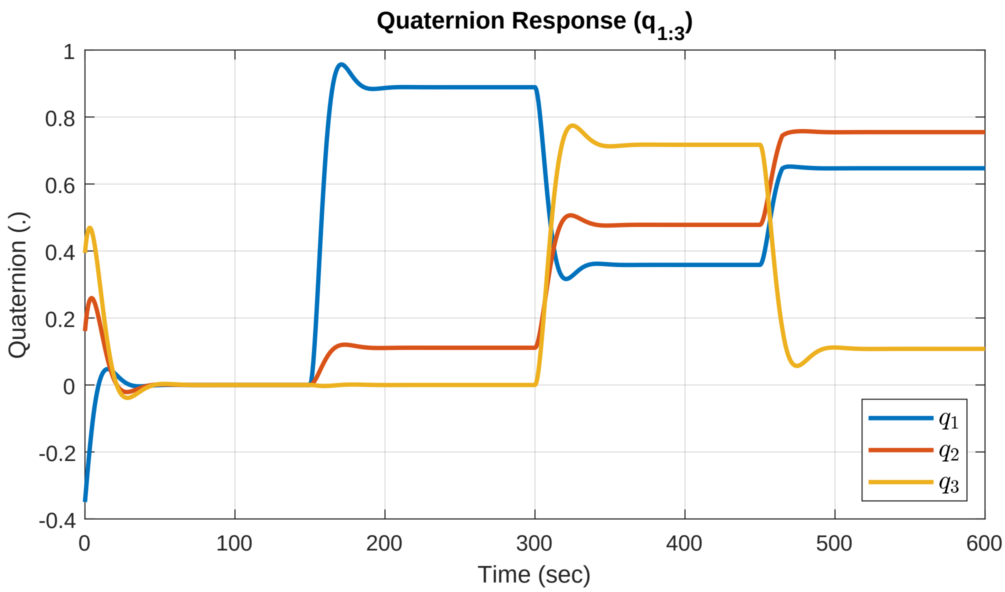


Figure 4.10: Closed-loop response of the satellite system with the matched uncertainties in (4.7), showing the attitude quaternion with no control input saturation.

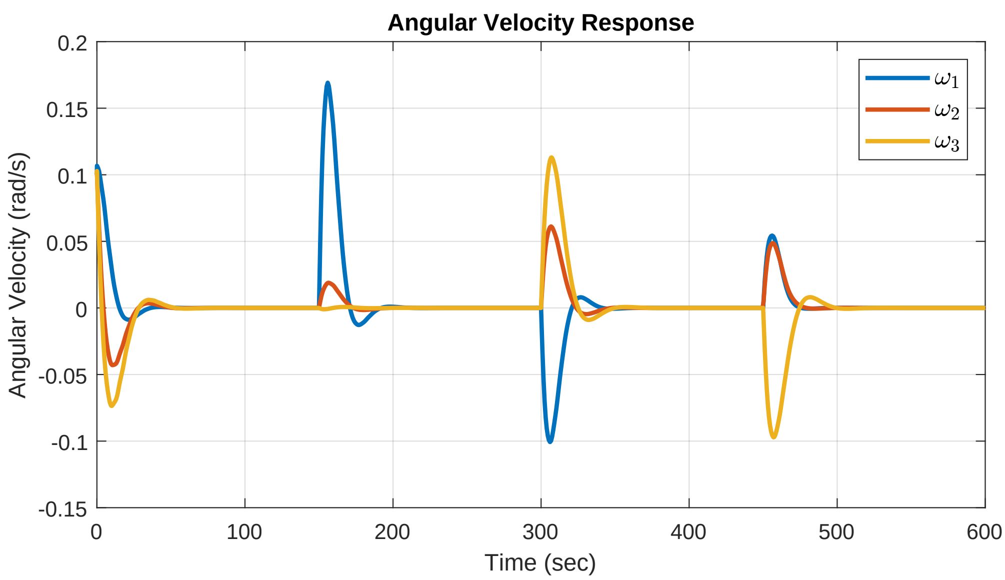
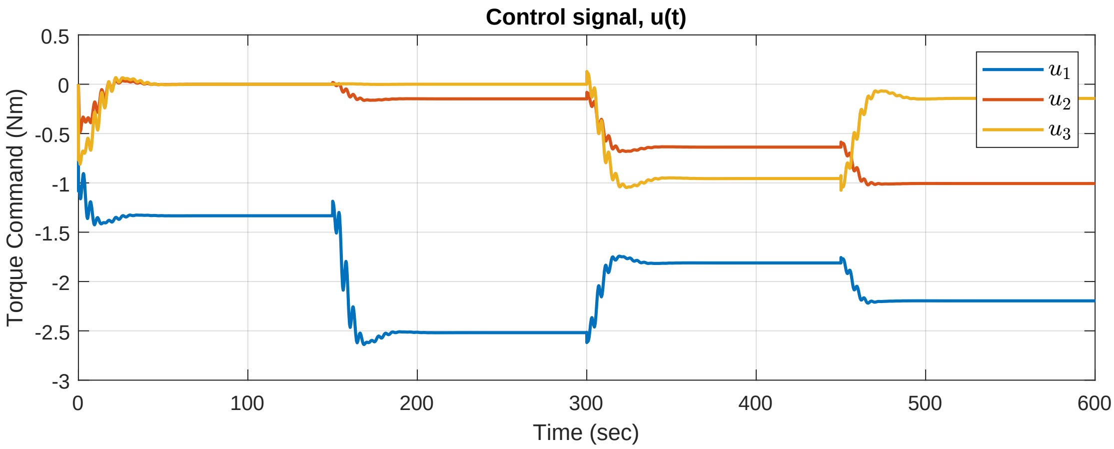


Figure 4.11: Closed-loop response of the satellite system with the matched uncertainties in (4.7), showing the angular velocity with no control input saturation.



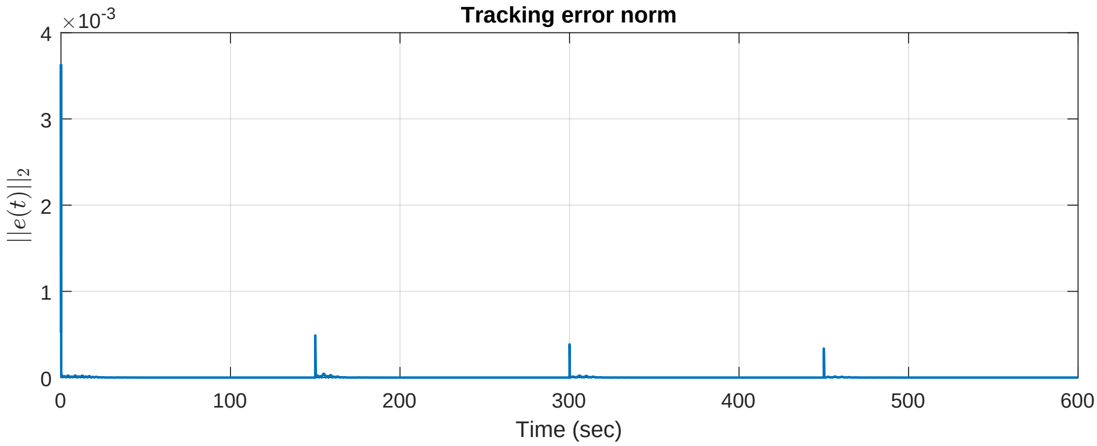


Figure 4.12: Control input with matched uncertainty without saturation.

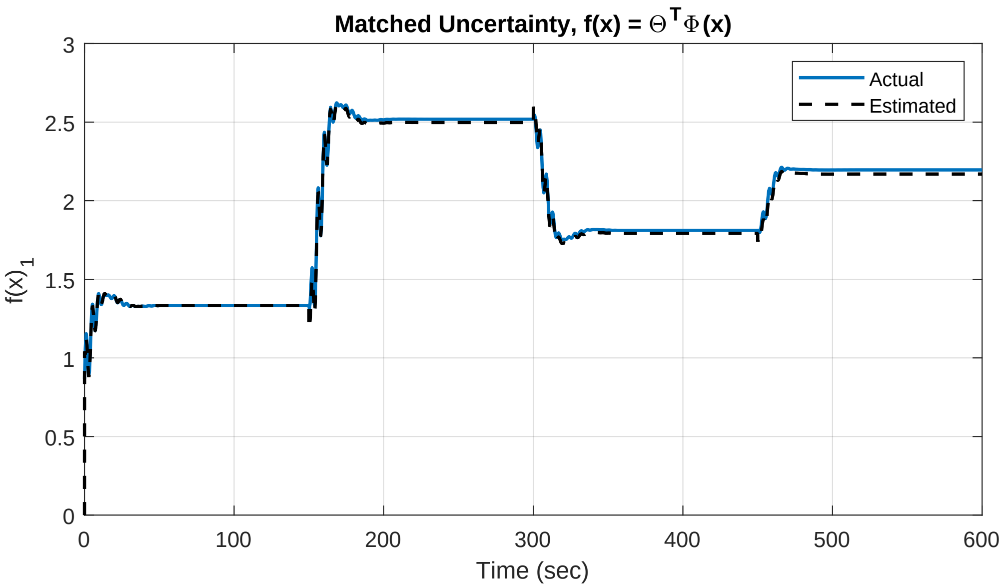


Figure 4.13: Plot of the first entry of f(x), showing great estimation of the actual applied uncertainty with no control input saturation.

Observing the responses in Figures 4.10 and 4.11 reveals that the addition of $\boldsymbol{K}_e$ has completely eliminated the transient oscillations. Bearing in mind of course, that the neglection of the control input saturation has improved the response and rapidness of the error convergence, as expected. In Figure 4.13, the matched uncertainty estimation is shown as the dashed line. Important to note, is that the uncertainty labeled 'Actual' has a different shape than that of Figure 4.7. This is simply because the uncertainty are state dependent and thus, as the response of the states change, so does $\Phi(x)$.

Without saturation of the control input a nearly perfect estimation of the uncertainty is achieved, and thus leads to its cancellation through the control term $-\hat{\boldsymbol{\Theta}}^T\Phi(x)$, hence the reasoning for observing it in the control signal.

However, the results of this implementation are invalid since the saturation has to be taken into account, but was included in the report merely to demonstrate how DMRAC handles uncertainty through estimation and cancellation. Therefore, a redesign is done to include the control input saturation as best as possible.

With the few following changes:

$$\boldsymbol{\Theta}^T = \begin{bmatrix} 1 & 0 & 0 & 1 & 0 & 0 & 1 \\ 0 & 1 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 1 & 0 \end{bmatrix} \quad \longrightarrow \quad \boldsymbol{\Theta}^T = \frac{1}{100}\begin{bmatrix} 1 & 0 & 0 & 1 & 0 & 0 & 1 \\ 0 & 1 & 0 & 0 & 1 & 0 & 1 \\ 0 & 0 & 1 & 0 & 0 & 1 & 1 \end{bmatrix} \tag{4.46}$$

$$\boldsymbol{Q} = \boldsymbol{I}_6 \quad \longrightarrow \quad \boldsymbol{Q} = \begin{bmatrix} Q_1 & 0 \\ 0 & Q_2 \end{bmatrix} \tag{4.47}$$

where

$$Q_1 = \begin{bmatrix} 0.1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 0.075 \end{bmatrix} \quad \wedge \quad Q_2 = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 0.35 \end{bmatrix} \tag{4.48}$$

the following response is obtained:

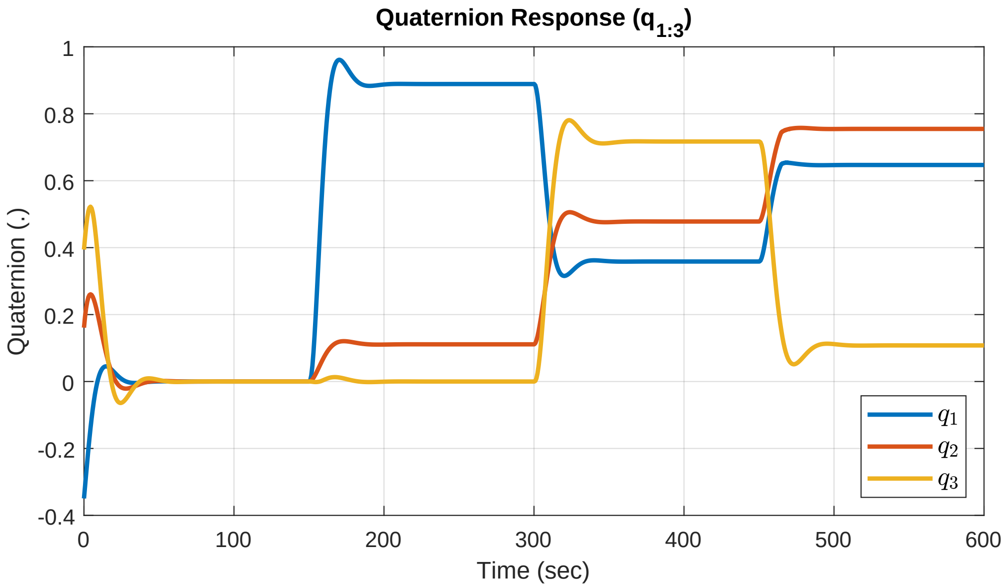


Figure 4.14: Closed-loop quaternion response with saturation of the control input.

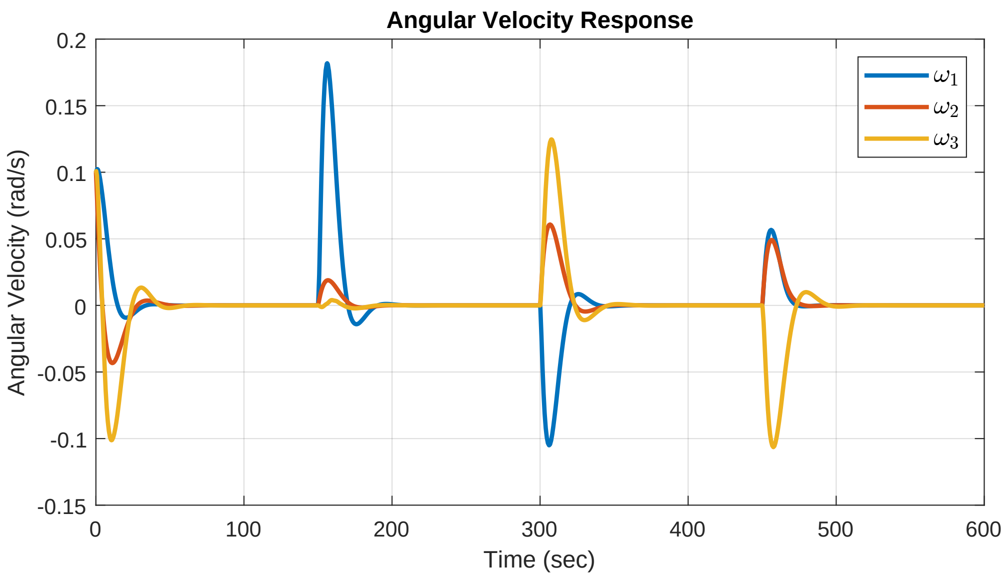


Figure 4.15: Closed-loop angular velocity response with saturation of the control input.

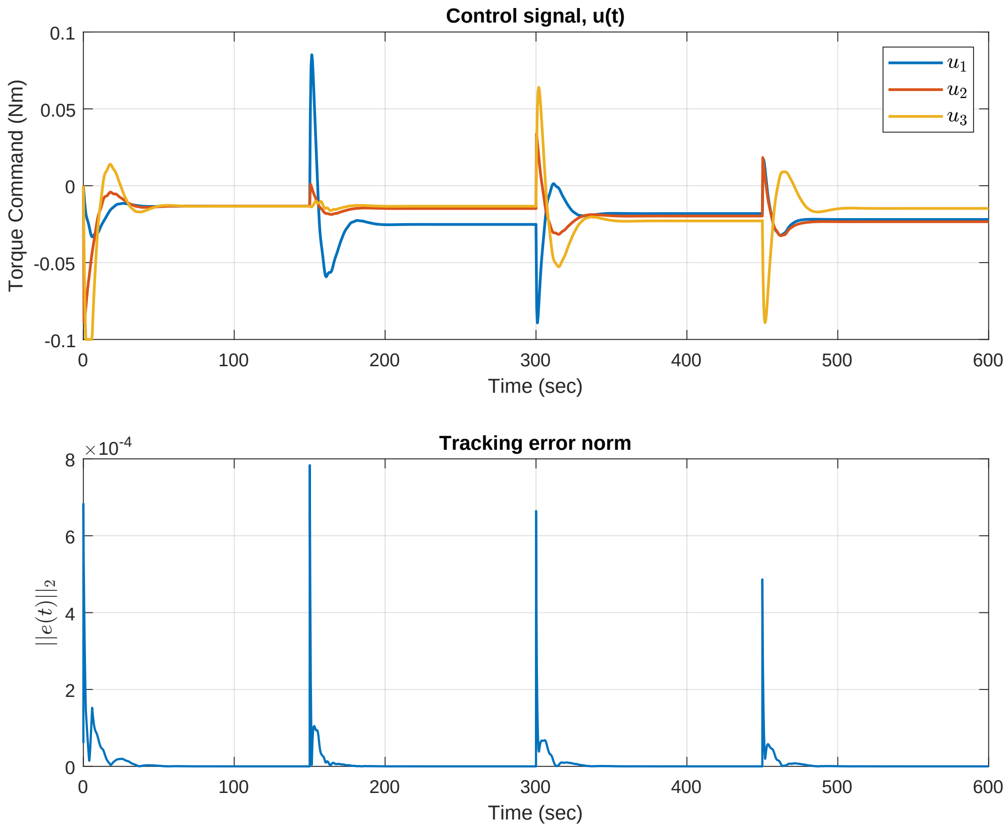


Figure 4.16: DMRAC control input staying withing the saturation limits of ± 0.1Nm, however, saturating very briefly in its initial transient (approximately four seconds).

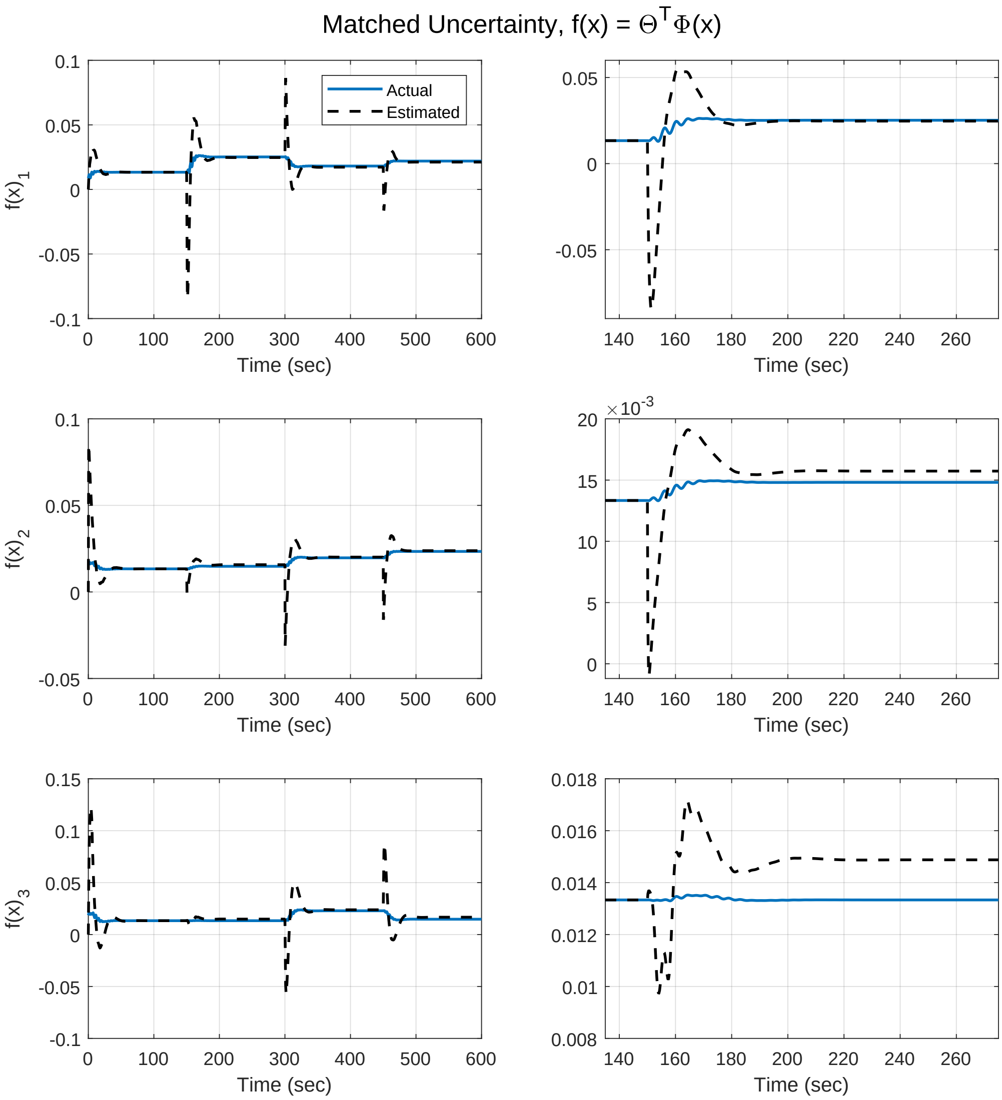


Figure 4.17: Plotted is both the system matched uncertainty $\mathbf{\Theta}^T\Phi(x)$ ('Actual') along with its parameter estimated $\hat{\mathbf{\Theta}}^T\Phi(x)$ ('Estimated'). First column is comprised of the full overview, and second column shows that same plot, however in a detailed view within a certain time interval.

One could argue that the solution of scaling $\mathbf{\Theta}^T$ is merely a convenient solution rather than an actual solution. As the quaternion in this case is a part of the basis function vector $\Phi(x)$, no other solution could be thought of, other than to scale down the problem. Observing the quaternion and angular velocity responses in Figure 4.14 and 4.15 however, no substantial change has occurred from the implementation without saturation, which is a plus.

Sadly in Figure 4.17, the $\hat{\mathbf{\Theta}}^T$ parameter estimation is unable to adapt fast enough to the now much smaller matched uncertainty $\mathbf{\Theta}^T\Phi(x)$ and therefore lacks greatly in the transients, moreover, an offset in the third vector entry of about 11% is observed, which does not support overly great estimation performance in this case.

Before rounding off the chapter, a short section comparing the response of the DMRAC in a nonlinear setting comprised of the satellite kinematics (2.17) and dynamics (2.22), to that of the response above, is performed to reveal the full extent of the DMRAC control design performed in this thesis.

DMRAC on the Nonlinear Dynamics

As may be apparent, the theory utilized throughout this chapter has been heavily guided by [7], and thus a linear state-space approach is used. This means that with the Lyapunov analysis performed earlier, one can assume that the Lyapunov stability would only guarantee tracking error convergence of the controlled linear system, therefore leaving no real guarantees for the nonlinear system. Because of that, a small comparison is made to reveal whether that assumption is true or not. With the same implementation method as in the previously shown pseudocode, however replacing the state-space dynamics with the nonlinear ones, yields the following:

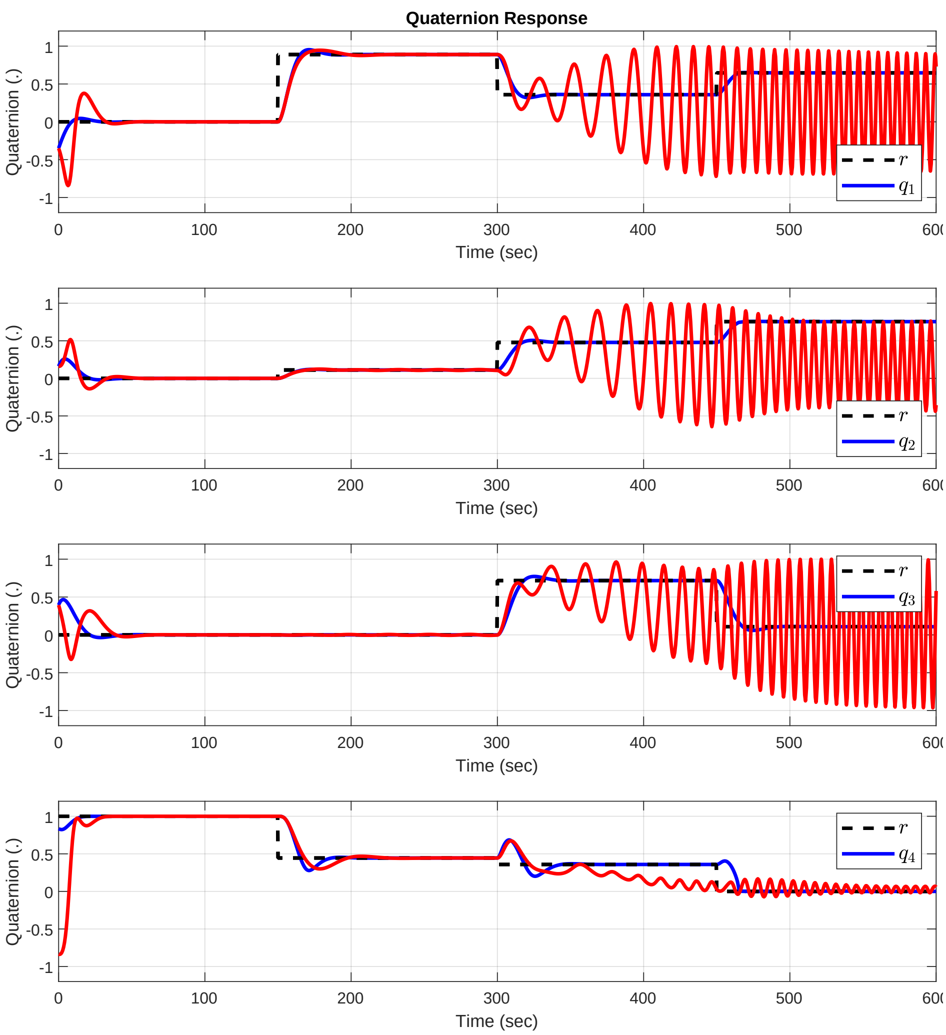


Figure 4.18: Comparison of the linear (blue) and nonlinear (red) closed-loop quaternion response under DMRAC control.

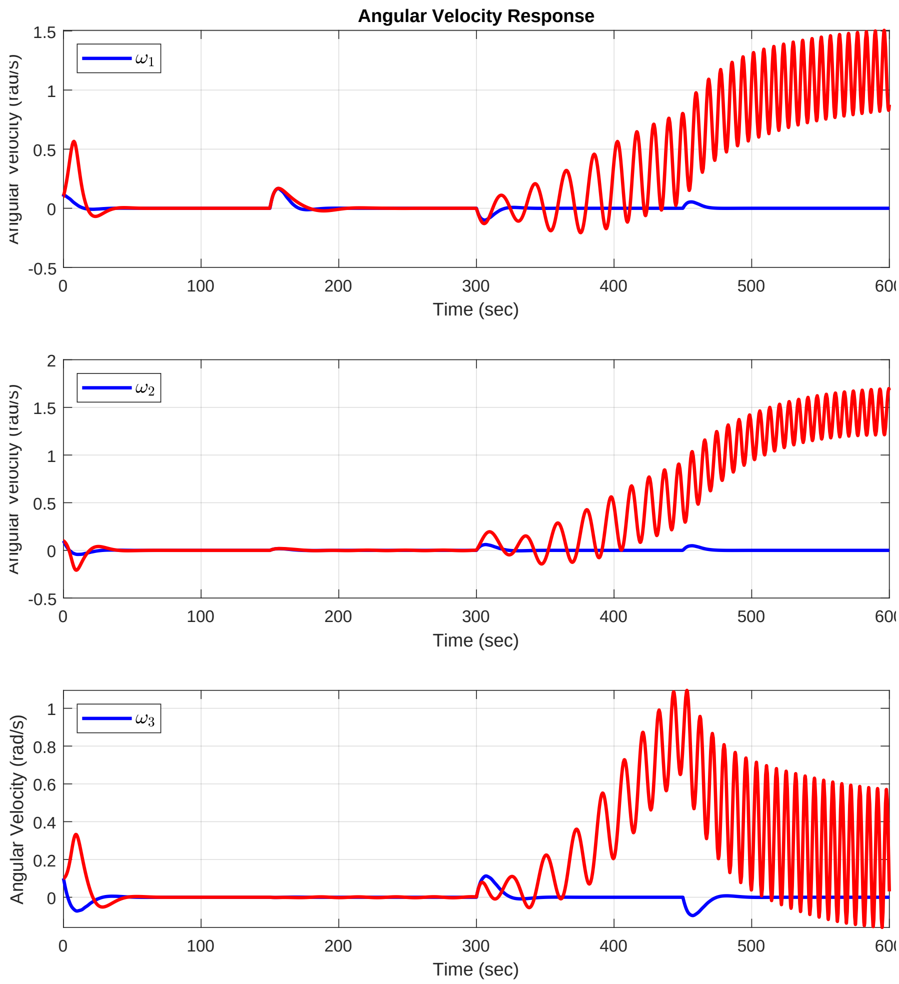


Figure 4.19: Comparison of the linear (blue) and nonlinear (red) closed-loop angular velocity response under DMRAC control.

Noting that the nonlinear system shown in red, follows the step references surprisingly well up until 300 seconds, where it starts to diverge. Because of previously stated assumption, it is no surprise to see the DMRAC controller not performing well on the nonlinear system without the proper analysis being made.

Hence a nonlinear system DMRAC design may be suggested for future work, along with other modifications to further improve DMRAC performance. All of which is discussed in the next section.

4.2 Future Work on MRAC

This section will be relatively short, working as a "mini discussion" about MRAC with some reflection on what modifications could be applied if one was to delve deeper into MRAC theory. It will be comprised of a discussion of its application possibilities, and a mentioning of some of the modifications to the MRAC approach.

Nonlinear System Application

At the time of writing, hardly any literature considers MRAC control of a nonlinear plant; with [7] among them. The reason as to why that is the case, has yet to be discovered. One assumption is of course, that nonlinear systems add more complexity to the control methods and convergence analyses, and hence using a linear representation simplifies that.

That being said, one brief section of [4] mentions the use of MRAC with a nonlinear plant in normal form, namely:

$$\dot{x} = f(x) + g(x)u \tag{4.49}$$

where the adaptable control input could be

$$u = \frac{1}{g(x)}[\hat{K}_f f(x) + \hat{K}_x x + \hat{K}_r r] \tag{4.50}$$

then following roughly the same steps as performed for the linear plant, one can end up with a DMRAC controller with a nonlinear plant. Noting that this example is not a perfect recitation of the one in [4], and merely used as a concept presentation.

Performance Modifications

For the DMRAC control design in Section 4.1.2 some modifications recommended by [7] can be utilized in future work, with the possibility to enhance the performance and robustness of the DMRAC controller. However, it is important to note that these methods are only being presented at a superficial level and rely solely on the recommendations and guidance of [7].

The Dead-Zone Modification

The dead-zone modification is a modification that enforces robustness of the adaptive laws facing unmatched disturbances, such as process noise, and is expressed as

$$\dot{\hat{\mathbf{\Theta}}} = \begin{cases} \mathbf{\Gamma}_\Theta \Phi(x) e^T \boldsymbol{PB}, & \text{if } \|e\| > e_0 \\ \mathbf{0}_{N\times m}, & \text{if } \|e\| \leq e_0 \end{cases} \tag{4.51}$$

which shows that if the norm of the tracking error gets within a certain value determined by e_0, the adaptive parameters no longer adapt and supposedly prevents the parameters from drifting, due to the process noise. Since (4.51) is discontinuous, a continuous approximation of the discontinuous dead-zone modification is instead suggested, which can be achieved with the following adaptive law:

$$\dot{\hat{\mathbf{\Theta}}} = \mathbf{\Gamma}_\Theta \Phi(x) \mu(\|e\|) e^T \boldsymbol{PB} \tag{4.52}$$

with

$$\mu(\|e\|) = \max\left(0,\, \min\left(1,\, \frac{\|e\| - \delta e_0}{(1-\delta)e_0}\right)\right), \qquad 0 < \delta < 1 \tag{4.53}$$

Providing what seems to be a strong tool for an MRAC control design for a system influenced by process noise and other unmatched disturbances of the sort.

Projection-Based Design

A projection-based MRAC design is suggested utilizing what they call the "projection operator" as a way of keeping the adaptive laws bounded and as way of preventing integrator windup. The integrators in question are the ones found in the feedback of the MRAC design, that results in the parameter estimates. For this design the adaptive laws take the following form

$$\dot{\hat{\boldsymbol{\Theta}}} = \mathrm{Proj}\Big(\hat{\boldsymbol{\Theta}},\, \boldsymbol{\Gamma}_\Theta \Phi(x) e^T \boldsymbol{PB}\Big) \tag{4.54}$$

where $\mathrm{Proj}(.)$ is the projection operator. This design makes use of chosen convex functions and convex sets based on the maximum allowed value of $\hat{\boldsymbol{\Theta}}$, namely $\hat{\boldsymbol{\Theta}}_{\max}$. If the adaptive parameter $\hat{\boldsymbol{\Theta}}$ lies within the defined convex sets, the projection operator does not perform any alterations to it. The mathematics behind the projection alterations and how it works is rather extensive mathematically, and will therefore not be covered here, but its result however, is shown for the curious individuals

$$\begin{aligned}\dot{\hat{\boldsymbol{\Theta}}} &= \mathrm{Proj}\Big(\hat{\boldsymbol{\Theta}},\, \boldsymbol{\Gamma}_\Theta \Phi(x) e^T \boldsymbol{PB}\Big)\\ &= \boldsymbol{\Gamma}_\Theta \begin{cases} (\Phi e^T \boldsymbol{PB}) - \dfrac{\boldsymbol{\nabla} f_\Theta \, \boldsymbol{\nabla} f_\Theta^T}{\|\boldsymbol{\nabla} f_\Theta\|^2_{\boldsymbol{\Gamma}_\Theta}} \boldsymbol{\Gamma}_\Theta (\Phi e^T \boldsymbol{PB}) f_\Theta, & \text{if} \quad \Big[f_\Theta > 0 \wedge (\Phi e^T \boldsymbol{PB}) \boldsymbol{\Gamma}_\Theta \boldsymbol{\nabla} f_\Theta < 0\Big] \\ (\Phi e^T \boldsymbol{PB}), & \text{if not} \end{cases}\end{aligned} \tag{4.55}$$

with f_Θ denoting the convex function mentioned earlier. Ending off the adaptive law modifications part.

Alternative Uncertainty Structures

In most cases, the type of matched uncertainty that is affecting the system will be completely unknown, and thus $\Phi(x) = x(t)$ is sometimes chosen, as also seen for the DMRAC implementation of this thesis. Naturally, this is a poor approximation of the uncertainty that one might see in practice and hence *universal approximators* (neural networks), can be utilized instead. Noting that these are used mostly for parametric uncertainty, such as system unknowns that were not accounted for in modelling, etc [7].

A feedforward neural network (NN) consists of an input, hidden layers containing neurons, followed by an output. Each neuron is a sum of all nodes feeding into it, multiplied by their respective weights and added biases. At the output all neurons are added and "activated" through nonlinear activation functions such as the Gaussian Radial Basis Function (RBF) and the Sigmoid function, both of which are showcased in [7]. An illustration of a feedforward NN is shown below.

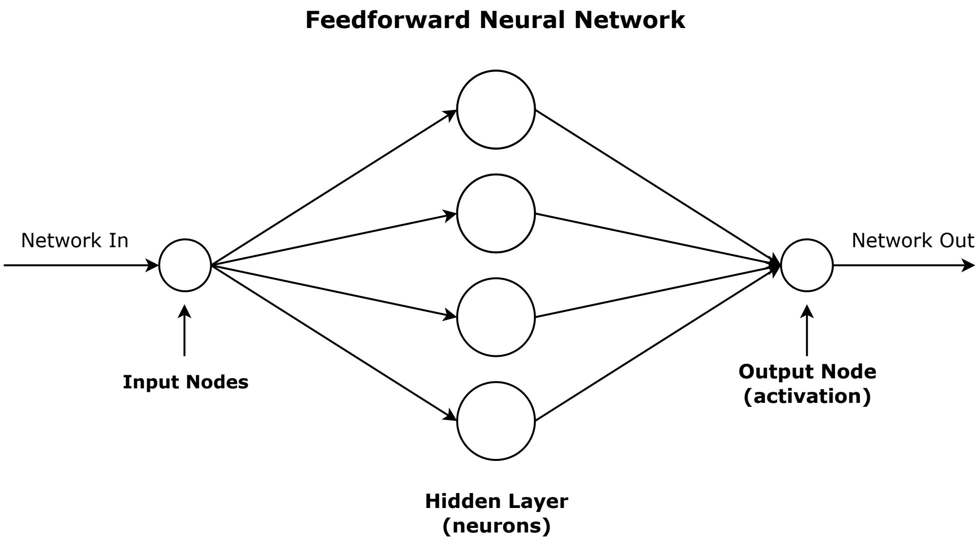


Figure 4.20: Feedforward neural network with one input, one output and four neurons.

Using both the activation methods mentioned above yields their respective feedforward NNs, briefly shown in the following.

Sigmoidal Feedforward NNs

This type of feedforward NN with *N* neurons is expressed as

$$NN(x) = W^T \vec{\sigma}(V^T x + \theta) + b \tag{4.56}$$

where $W \in \mathbb{R}^{N\times m}$ is the matrix of "outer-layer" weights and $V^T x + \theta$ representing a system of linear equations, describing all the inner weights and biases that are to be activated through the Sigmoid function $\sigma(x)$, where $V \in \mathbb{R}^{n\times N}$, $\theta \in \mathbb{R}^N$ and lastly $b \in \mathbb{R}^m$ denoting the NN bias.

Feedforward RBF NNs

In the case of the RBF neural network, the following is obtained

$$NN(x) = \theta^T \begin{pmatrix} \varphi\big(\|x - C_1\|_{W_1}\big) \\ \vdots \\ \varphi\big(\|x - C_N\|_{W_N}\big) \end{pmatrix} + b = \underbrace{\begin{pmatrix} \theta^T & b \end{pmatrix}}_{\mathbf{\Theta}^T} \underbrace{\begin{pmatrix} \varphi_1(x) \\ \vdots \\ \varphi_N(x) \\ 1 \end{pmatrix}}_{\Phi(x)} = \mathbf{\Theta}^T \Phi(x) \tag{4.57}$$

with the parameters stated as $\mathbf{\Theta} = (\theta^T \quad b) \in \mathbb{R}^{(N+1)\times m}$, the vector of weights, $C_i \in \mathbb{R}^n$ is the center of the i^{th} Gaussian, $W_i = W_i^T > 0$ is the matrix weighting the norms within $\varphi_i(x)$, again b being the NN bias and lastly $\Phi(x) \in \mathbb{R}^{N+1}$ is the regressor vector with all but last entry comprised of the RBF activation function $\varphi_i(x) = \varphi\Big(\|x - C_i\|_{W_i}\Big)$.

As an way to wrap up the section, utilizing universal approximators would yield a DMRAC control input of the following form

$$u = \hat{\boldsymbol{K}}_x \, x + \hat{\boldsymbol{K}}_r \, r - \hat{f}(x) \tag{4.58}$$

since NNs are universal approximators, they can approximate any function and thus assumed to be able to mitigate the vast majority of parametric uncertainties there might arise. Therefore hopefully satisfying $\Delta f(x) = \hat{f}(x) - f(x) \rightarrow 0$ as $t \rightarrow \infty$, with $f(x)$ being the matched uncertainties of (4.1). And with this bringing an end to the future work section, which leaves the conclusion of the thesis on whether or not the goals have been met.

Chapter 5

Conclusion

PD/LQR

In conclusion, the findings of this thesis demonstrate that the controller based on the Principal axis does not outperform the controller based on the Body frame in both PD and LQR implementations, despite their closely matched performances. However, when comparing the LQR and PD controllers implemented with the Body Frame, it was observed that the LQR controller exhibits a noticeably better performance. The observed difference in performance between the PD and LQR controllers can be boiled down to a matter of tuning. By finding more appropriate gains for the PD controller, it is possible to achieve a response that closely resembles that of the LQR controller.

Moreover, all four cases when including variations in the inertia matrix performed satisfactorily, with the control signal staying within the reactions wheels saturation limits. Thus it is possible to conclude that the objective of developing a controller based on the Principal Axis of Inertia and Body Frame and performing a comparison between both, as proposed by Space Inventor, was successfully achieved and the resulting findings were presented.

Model Reference Adaptive Control

In this study, a successful implementation of an MRAC (Model Reference Adaptive Control) scheme is achieved, which effectively tracks a desired reference model. Despite utilizing only the basic theory of DMRAC (Direct MRAC), the adaptation and cancellation of matched uncertainty is successfully demonstrated, however struggling a bit when having to keep within the saturation limits. Furthermore, the implemented uncertainty structure $\Phi(x)$ falls short in comparison to the potential offered by the presented neural network structures in Section 4.2 when the structure of the matched uncertainty is completely unknown.

The DMRAC simulation results obtained from the nonlinear case reveal that the system is capable of accurately tracking references that are close to the operating points. However, when the references deviate significantly from these points, the system becomes unstable. This observation indicates the need for further investigation to understand the underlying causes leading to this instability as stated in Subsection 4.1.3.

On a last note, additional study on the topic of adaptive control scheme is required to gain a deeper understanding of the MRAC implementation and identify potential solutions or improvements such as the ones mentioned in Section 4.2.

Bibliography

[1] Bernard Friedland. *Control System Design: An Introduction to State-Space Methods.*

[2] F. Landis Markley, John L. Crassidis. *Fundamentals of Spacecraft Attitude Determination and Control.*

[3] *Gravity Space Website.* https://gravity-space.com/. Accessed: 31-05-2023.

[4] Petros Ioannou and Jing Sun. *Robust Adaptive Control.* Dover Books on Electrical Engineering. Dover Publications, 2012. ISBN: 978-0486498171.

[5] Hassan K. Khalil. *Nonlinear Control, Global Edition.* Pearson Education Limited, 2015. ISBN: 9781292060507.

[6] Laurids Hovgaard. *Danish satellite manufacturer to send the world's first nanosatellite into geostationary orbit.* https://ing.dk/artikel/danish-satellite-manufacturer-send-worlds-first-nanosatellite-geostationary-orbit. Accessed: 31-05-2023.

[7] Eugene Lavretsky and Kevin A. Wise. *Robust and Adaptive Control.* Advanced Textbooks in Control and Signal Processing. Springer London, 2012. ISBN: 978-1-4471-4396-3.

[8] Bellar A; Si Mohammed MA. "Satellite Inertia Parameters Estimation Based on Extended Kalman Filter". In: *Journal of Aerospace Technology and Management* 11: e1619 (2019). DOI: https://doi.org/10.5028/jatm.v11.1016.

[9] F. Landis Markley and John L Crassidis. *Fundamentals of Spacecraft Attitude Determination and Control.* Springer, 2014. ISBN: 9781493908011.

[10] NASA. *The Evolution of CubeSat Spacecraft Platforms.* https://ntrs.nasa.gov/citations/20210021348. Accessed: 05-04-2023.

[11] NASA. *What are SmallSats and CubeSats?* https://www.nasa.gov/content/what-are-smallsats-and-cubesats. Accessed: 04-04-2023.

[12] J. David Powell, Gene F Franklin, and Abbas Emami-Naeini. *Feedback control of dynamic systems.* eng. Pearson Education, 2019. ISBN: 1292274522.

[13] Marcel J. Sidi. *Spacecraft Dynamics and Control: A Practical Engineering Approach.* Cambridge Aerospace Series. Cambridge University Press, 1997. DOI: 10.1017/CBO9780511815652.

[14] Wen Tan, Wenjie Han, and Jiaohu Xu. "State-space PID: A Missing Link between Classical and Modern Control". In: *IEEE Access* PP (Jan. 2022), pages 1–1. DOI: 10.1109/ACCESS.2022.3218657.

[15] Brian Gasberg Thomsen and Jens Nielsen. "CubeSat Sliding Mode Attitude Control - Developing Testbed for Verification of Attitude Control Algorithms". Master's thesis. Aalborg University, 2016.

Appendix A

Space Inventor Project Proposal

Accurate Three-Axis Control of Spacecraft with Non-Uniform Mass Distribution

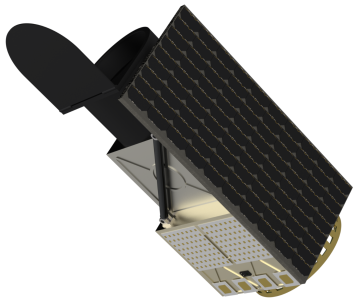
Space Inventor, an innovative satellite designer and manufacturer in Aalborg, is looking for engaged students who desire to solve real world problems within the Space Industry. The project proposal is intended to be executed in the form of a master-level semester project.

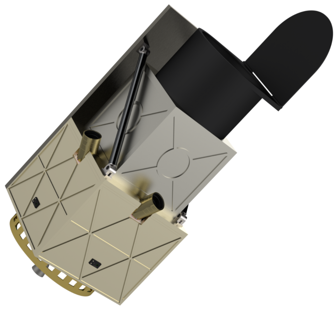
Problem Description

The Space Industry is experiencing exponential growth in the interest from private and governmental customers who desire to initiate space missions ranging from science-based experiments to better understand our world and universe, to commercial communication projects or measurement services. Common to all are that with the current state of technology, small and low-cost satellites can be made while still providing high-precision data for customers.

For many modern antennas and optical payloads, ultra-fine pointing precision is required of the spacecraft. To accomplish this, it is key that the highest attitude estimation accuracy can be achieved with the on-board sensors including magnetometers, gyroscopes, fine sun sensors, and star trackers. To that end, the spacecraft controller needs to be able to control the three body axes of the spacecraft. Typically it is assumed that the inertia matrix of the satellite is diagonal, meaning that performing control on either of the x, y, and z body axes of the spacecraft will not cause a coupling to the other axes. For practical cases, however, the inertia matrix of the spacecraft always contains non-diagonal non-zero terms that contribute to cross-coupling between the satellite axes. For microsatellites with large propellant tanks, the mass distribution of the spacecraft can be very non-homogeneous, resulting in undesired cross-coupling dynamics on the spacecraft attitude. Typically a PD, PID, or LQR controller is implemented as an attitude controller on the satellite body axes. However, if the mass is not homogeneously distributed amongst the three axes (non-diagonal inertia matrix), the controller should somehow compensate for this by translating the control problem into the principal-axes coordinate system (where the inertia matrix is diagonal) instead of the satellite body frame (where the inertia matrix is non-diagonal).

In the context of spacecraft dynamics and control, we distinguish between Attitude Control system (ACS) which concerns commanding actuators, and Attitude Determination System (ADS), which concerns the estimation of the current orientation of the satellite. The aim of this project is to review the state of the art of performing spacecraft control in the principal-axis control reference frame and propose/design a method that will be applicable for actual spacecraft operation.





Project formulation:

How should satellite attitude control be performed in the principal-axes control reference frame to eliminate/reduce the cross-coupling dynamics resulting in a non-diagonal satellite inertia matrix?

Verification on Space Inventor’s High-Fidelity Satellite Simulator

It will be possible to test the proposed estimation algorithm in Space Inventor’s high-fidelity satellite simulator to verify your own simulation as well. This, however, requires that the algorithm/control is developed in C language. Alternatively, a simulation study can be made using university simulation licences directly instead of verification on Space Inventor’s simulator.

Prerequisites

It is expected that the student is on the master level of a relevant engineering or mathematical school with intermediate knowledge and understanding in dynamical systems, control system theory, engineering mathematics and statistics, and filter design. A background knowledge of satellites, quaternion representations, and the employed sensors is recommended but not a requirement.
It is an advantage to have some experience with C-programming.